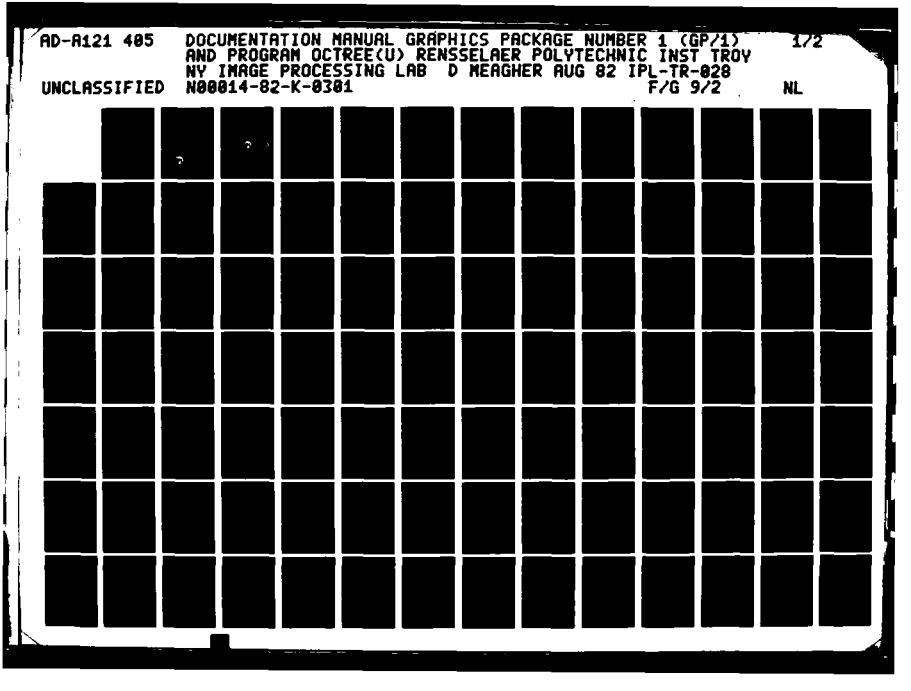
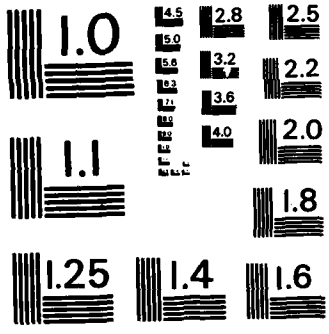

AD-A121 405 DOCUMENTATION MANUAL GRAPHICS PACKAGE NUMBER 1 (GP/1) 1/2
AND PROGRAM OCTREE(U) RENSSELAER POLYTECHNIC INST TROY
NY IMAGE PROCESSING LAB D MEAGHER AUG 82 IPL-TR-028
UNCLASSIFIED N00014-82-K-0301 F/G 9/2 NL



(12)

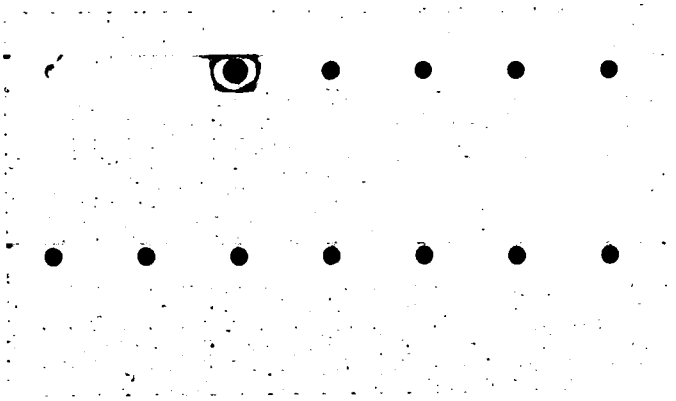


Image Processing Laboratory

Rensselaer Polytechnic Institute
Troy, New York 12181

(12)

IPL-TR-028

Documentation Manual
Graphics Package No. 1 (GP/1)
and
Program OCTREE

Donald Meagher

August 1982

N00014-82-K-0301

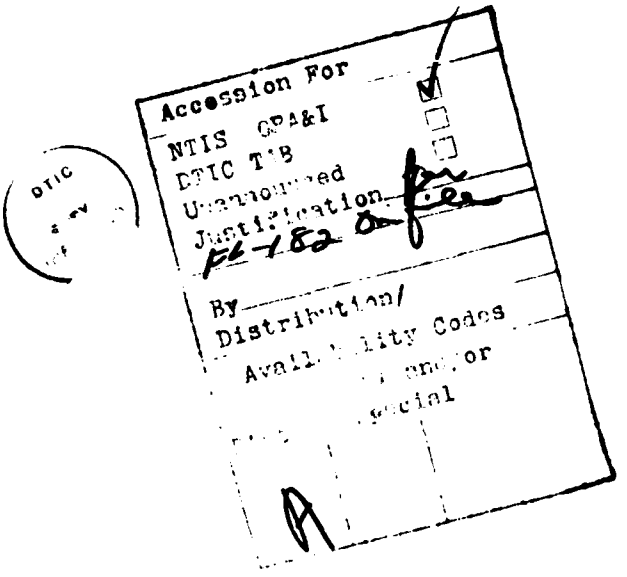
Image Processing Laboratory

Electrical and Systems Engineering Department
Rensselaer Polytechnic Institute, Troy, New York 12181

ABSTRACT

Graphics Package No. 1 (GP/1) is a general-purpose support package for graphics applications. It is written in Fortran and operates on a Prime computer system with an Imlac 6220 graphics terminal. The user interface is presented along with a sample work session. It is intended for users of systems based on GP/1. Programmer information is also provided for new applications.

Program OCTREE is a solid modeling package based on GP/1. It also requires a DeAnza IP5532 color raster graphics display for shaded image output (the system was developed for use at the Image Processing Laboratory at Rensselaer Polytechnic Institute). User and programming information is presented.



ACKNOWLEDGEMENT

The programs documented in this report were developed by the author under the direction and support of Professor Herbert Freeman of the Electrical, Computer and Systems Engineering Department and Director of the Image Processing Laboratory (IPL) at Rensselaer Polytechnic Institute.

The work was funded in part by the National Science Foundation's Automation, Bioengineering and Sensing Program under grant ENG-79-04821 and in part by the Office of Naval Research under Contract N00014-82-K-0301, NR049-515.

Table of Contents

1 Introduction

1.1 Scope

This manual describes and documents Graphics Package No. 1 (GP/1), a general-purpose, high-level applications support package and program OCTREE, a solid modeling application based on GP/1. It is intended to accompany a magnetic tape containing the GP/1 and OCTREE systems.

GP/1 consists of an information file, a main program, an insert file, and 79 subroutines containing 6,274 lines of Fortran code. Program OCTREE is composed of an information file, a main program, five insert files and 97 subroutines. It contains 20,415 lines of Fortran code.

1.2 Operating Environment

Both GP/1 and program OCTREE were developed for use on a Prime 750 computer with an Imlac 6220 graphics terminal. For display of shaded surface objects, a DeAnza IP5532 color raster graphics display is needed.

1.3 Related Documents

The reader should be familiar with the following manuals:

-, "The Fortran Programmer"s Guide," Prime Computer, Inc., FDR3057

Smith, K., "User"s Manual - Interactive Computer Graphics Center," Rensselaer Polytechnic Institute

The following documents will be of interest to those planning to use or modify program OCTREE:

Meagher, D., "Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer," Technical Report IPL-TR-80-111, Image Processing Laboratory, Rensselaer Polytechnic Institute, October 1980

Meagher, D., "Geometric Modeling Using Octree Encoding," Technical Report IPL-TR-81-005, Image Processing Laboratory, Rensselaer Polytechnic Institute, February 1981

Meagher, D., "Octree Generation, Analysis and Manipulation," IPL-TR-027, Image Processing Laboratory, Rensselaer Polytechnic Institute, April 1982

1.4 Overview

Section 2 provides an introduction to GP/1 for the new user. A demonstration GP/1 system has been implemented and is contained on the associated tape. The individual functions are documented in Appendix A. A step-by-step work session using the demonstration system is presented in Appendix B.

Section 3 presents user information for program OCTREE.

Sections 4 and 5 contain programming information. Section 4 is intended for those wishing to develop new applications based on GP/1. Section 5 provides information necessary to further develop program OCTREE or to implement systems based upon it.

2 Graphics Package No. 1 (GP/1)

Graphics Package No. 1 (GP/1) is a user-oriented, general-purpose graphics package designed to free the applications programmer from the lower-level details of graphics programming. To the user, it provides a standardized screen format and interactive environment over a variety of applications.

The user interface is through an Imlac 6220 vector refresh graphics terminal. The primary interactive device is the lightpen. Alternate devices are a keyboard, arrow keys, two sets of function buttons and a graphics tablet. Figure 2.1 shows the screen format. Screen elements are as follows:

Workspace - A 9 inch by 9 inch workspace (large central area enclosed by square in Figure 2.1) contains all graphics. The center of the workspace is the center of the GP/1 coordinate system (x to the right, y up and z into the screen).

Function Keys - A set of 32 permanently defined (for a particular application) function "keys" numbered 0 to 31 are located to the left and right of the workspace. They are activated by lightpen selection. Two are permanently defined over all applications and are always active whenever the lightpen is active. They are key 15, 'HD COPY' (generate a hardcopy of whatever is on the screen) and key 31, 'EXIT' (terminate session if

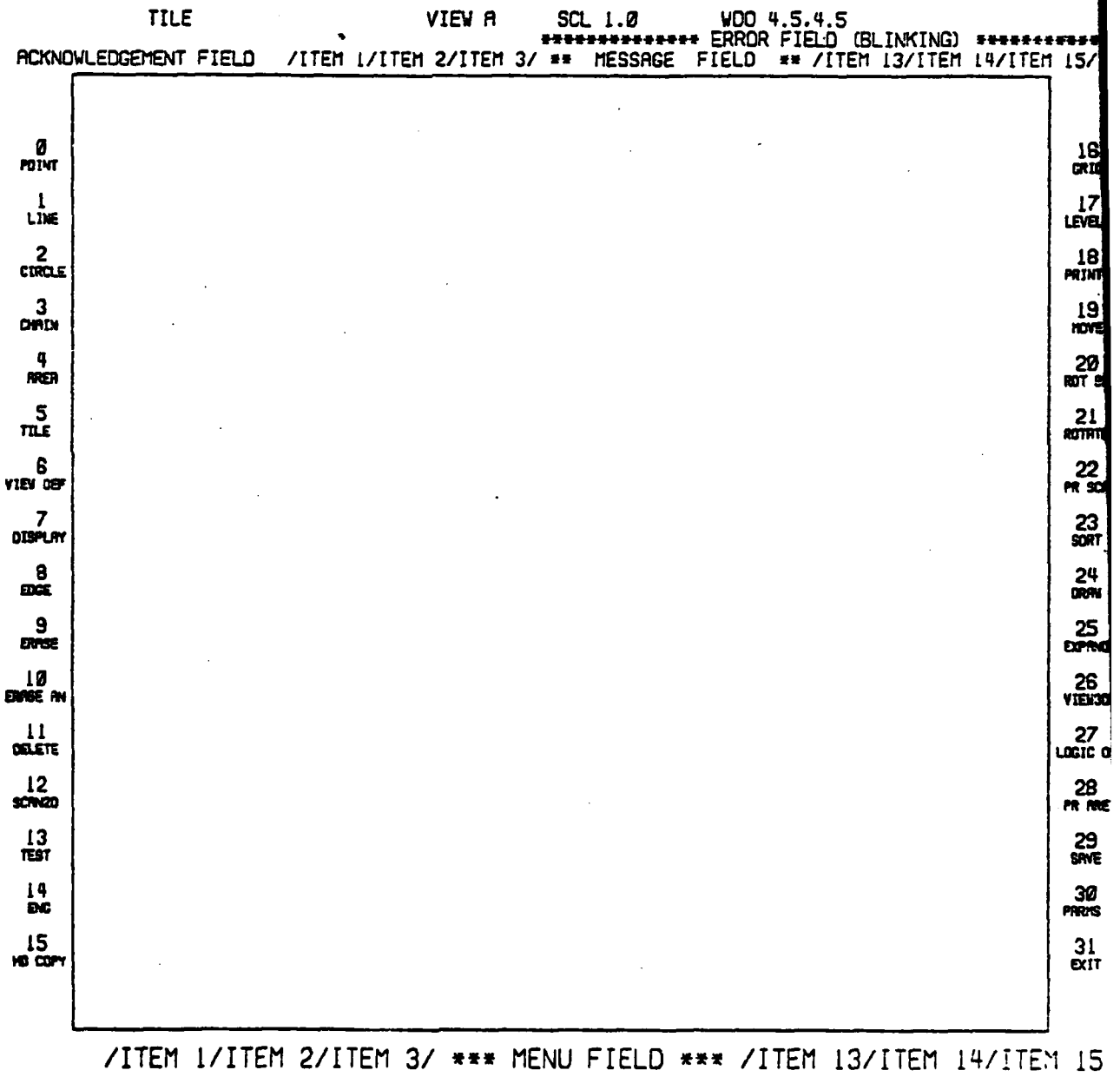

TILE
VIEW A
SCL 1.0
WDO 4.5.4.5
ERROR FIELD (BLINKING)
ACKNOWLEDGEMENT FIELD /ITEM 1/ITEM 2/ITEM 3/ ** MESSAGE FIELD ** /ITEM 13/ITEM 14/ITEM 15/
0 POINT
1 LINE
2 CIRCLE
3 CHAIN
4 AREA
5 TILE
6 VIEW DEF
7 DISPLAY
8 EDGE
9 ERASE
10 ERASE AN
11 DELETE
12 SCANZO
13 TEST
14 ENG
15 MB COPY
16 GRI
17 LEVEL
18 PRINT
19 MOVE
20 ROT S
21 ROTATE
22 PR SC
23 SORT
24 DRAW
25 EXPAND
26 VIEW3D
27 LOGIC O
28 PR ARE
29 SAVE
30 PARMS
31 EXIT
/ITEM 1/ITEM 2/ITEM 3/ *** MENU FIELD *** /ITEM 13/ITEM 14/ITEM 15

Figure 2.1

selection is then confirmed by the user).

Current Function - The field in the upper left corner of the screen displays the most recent function key selected (key 'TILE' in Figure 2.1).

Acknowledgement - The left side of the third line (from the top) is used to acknowledge user action.

Message Field - The right side of the third line can contain 1 to 15 individual message items. They are typically separated by the slash character ('/'). They are individually selectable using the lightpen.

Menu Field - The field at the bottom of the screen presents 1 to 15 lightpen selectable items to the user. The message field at the top of the screen is typically used for the selection of desired operations and actions while the menu field is usually used for selecting particular numbered objects (if displayed, objects can usually also be selected directly from the screen using the lightpen).

Error Messages - The second line presents a blinking error message to the user. The text is right justified. The posted error message is cleared if the error field is selected with the lightpen.

The GP/1 system provides convenient facilities for the initialization of the screen and system tables, for the storage, entry and retrieval of object elements (points, lines, circles, arcs, chain encoded curves and user defined

elements), for control of interactive devices, for maintenance of the screen, to perform geometric operations (eg., compute the intersection of two lines), handle files, and so on.

An applications library is available. It contains a number of application related routines (generation of a fillet, a Bezier curve function, etc.).

In order to acquaint the new user with the GP/1 package, a demonstration version of the system has been implemented. The various functions accessed from the function keys are either basic GP/1 functions or were selected from the applications library. Those function key definitions are presented in Appendix A. A step-by-step sample work session is contained in Appendix B.

3 Program OCTREE

Program OCTREE is an application program based on GP/1. It was implemented to develop, verify and demonstrate the Octree Encoding method of solid modeling. It is assumed that the reader is familiar with the octree techniques (see references listed in Section 1). This section will present the program from the viewpoint of a user.

The following function keys remain unchanged from the previous section: 0, 1, 2, 3, 7, 9, 11, 15, 17, 30 and 31. The remaining keys perform various quadtree and octree entry, manipulation and display functions.

In the current implementation, up to 10 chain coded objects (chains 1 to 10) can be defined at one time. Likewise, up to 10 quadtree (2-D) or octree (3-D) objects can exist at one time (objects 1 to 10). All can be erased from the screen (if currently displayed) using key 9. They can be deleted (completely eliminated from the system) by selecting key 11. Chains and quadtrees (but not octrees) can be displayed using key 7.

When a chain or object is deleted, its table entries are cleared and the link or node space recovered for future use. The chain number or object number is made available for use in later operations.

Chains are entered by employing key 4. The appropriate level should first be selected using key 17. If the chain

code is not returned to the starting point by the user (to form a closed curve) this is performed by the system.

Quadtrees are entered into the system by first entering chain coded curves (see pages A-14 to A-18 in Appendix A) and then converting one or more using key 4. Chains should be entered in a clockwise direction for exterior boundaries or counter-clockwise for interior holes (interior of 2-D object always to the right). Key 4 operation is as follows:

KEY 4: QUADTREE - When selected, a chain coded closed curve can be selected using the lightpen either directly from the screen (if currently displayed) or by selecting its chain number from the menu field. If multiple chains are to be used to define the object (allows disjoint parts or interior holes), the option '/MULTIPLE CHAINS/' should be picked. Multiple chains can now be selected. The item '/END/' in the menu field should be picked after the last chain has been requested. There is no significance to the order of chain entry.

Two additional options may be selected. Option '/NEW LEVEL/' allows the user to request that the quadtree be generated at a level other than the current level. Option '/INTENS/' can be used to change the screen intensity (brightness) of the resulting quadtree. Both options must be selected before chain entry (single chain) or before terminating chain entry (multiple chains).

After option and chain selection, the quadtree is generated and displayed. Its object number is displayed in the message field. If no object numbers are available, the operation is aborted with no action taken.

Several methods are available for 3-D octree entry. They are currently as follows:

(1) generating a prism from a quadtree (keys 6 & 13)

(2) direct generation of the octree (eg., key 5)

(3) generation from an application function (eg., NC verification function)

(4) entry from disc file generated manually or by an auxiliary program (conversion from binary images, etc.)

Relevant octree generation keys are the following:

KEY 6: EXPAND - The 'expand' function creates a prism from a quadtree. The prism extends in the z direction (into the screen) for the entire depth of the universe. The number of nodes generated for higher precision quadtrees (lower levels) can be prohibitive. Care should thus be exercised when using this function.

The expansion function is typically used in conjunction with 90 degree rotation and object intersection. Thus, for example, the three orthogonal views of an object can be entered, converted to prisms, rotated appropriately and then the intersection of all three taken to generate an object. It should be noted that only a limited class of objects can be generated in

this way.

In operation, a quadtree is selected either directly from the screen or by selecting its object number. If an input object is of dimensionality other than two (ie., not a quadtree) the order of the object must be indicated. A "hyperprism" of one greater dimensionality will be created. After generation, the object number of the new object is presented in the message field.

KEY 5: CIR CH - (Circular Chain) - This key allows the entry of a circular chain or the direct generation of an octree sphere. When selected, the user picks '/CIRCLE CHAIN/' to generate a 2-D circular chain or '/SPHERE/' to generate a 3-D sphere. In either case, the center in the x-y plane (the screen) is entered using the lightpen followed by the entry of a point on the edge or surface. In the case of a sphere, the z value of the center is entered using the lightpen. The x value (positive to the right) indicates the z value (depth behind screen).

After generation, the circular chain is displayed or the object number of the sphere is indicated.

KEY 13: SLICE - This key allows the entry of a prism not extending throughout the universe. If a quadtree is selected without options, it is given a depth (in z) corresponding to the width of a node at the current level (this is called a slice). The user is requested to enter the location of the slice in z. This

is entered with the lightpen (positive z to the right).

If option '/STACK/' is selected, two or more slices at increasing values of z are unioned together. This generates a prism with a depth greater than that of a single slice. The starting point and ending point in z are entered to define the exact locations. If this option is picked, the option '/OFFSET/' becomes available. This allows a number of empty slices to be placed between solid slices.

The following keys are used to manipulate quadtrees and octrees:

KEY 8: EDGE - This function will display only the edges of a quadtree or, for a quadtree or octree, generate a new object that is the "visible shell" of the objects (F nodes not visible when using the display program are converted to E). To display the edges, the quadtree is simply picked with no options specified. To generate a shell for a quadtree, '/SHELL 2D/' is selected. For an octree shell, '/SHELL 3D/' is picked. If a shell is generated, the new object number is displayed.

KEY 19: MOVE - This key will move (translate) an object (2-D or 3-D). The user should first select the desired minimum level for the translated object. If none is picked, the current level will be used. If the object is 3-D, the option '/3D/' should be selected.

The quadtree is picked directly from the workspace or through its object number. The translation vector is then requested in two or three dimensions. The new object number is displayed.

KEY 22: SCALE - This function key is used to generate a scaled version of an object. Level and object selection is similiar to that for key 19. A new origin can be entered as well as a scale factor (1.0 to 2.0) in each of two or three dimensions.

KEY 20: ROT 90 - This function generates a new object that is the selected object rotated by 90 degrees. The user is given various options for generating a new quadtree, including reflection.

For octrees, the option '/3D/' should be selected. A choice of rotation about x, y or z is offered. For x and y, the object is rotated CCW when viewed from a point on the axis in the positive direction. For rotation in z, the rotation is CW so that the rotation is CCW for the user (the user is in the negative z direction relative to the screen).

KEY 21: ROTATE - This allows the rotation of an object by an angle between 0 degrees and 90 degrees. Selection is similar to that for key 19 except that a center of rotation is requested and a rotation angle (0 to 90 degrees) is entered. For an octree, the rotation is only about the z axis. For a completely general 3-D

rotation, 90 degree rotation should be used in conjunction with three rotations about the z axis.

KEY 27: SET OPS - This key allows the user to generate a new object which is a Boolean combination of two objects. The user first selects the operation of union, intersection or difference (the first option, 'MATCH' is an early form of quadtree intersection).

The user then selects two objects. If they are other than 3-D, this should be indicated with a lightpen selection. The number of the new object is displayed.

Utility functions are as follows:

KEY 28: TABLES - This function allows the user to examine various internal tables. Of particular interest are the following:

/OBJECT/ - Display all available information on the current objects.

/MODIFY/ - Selected object parameters can be modified. Note: This option should only be employed with great care. The inadvertent change of an object parameter could cause undesired results. No checks are made to determine if a modification is appropriate.

/NODES/ - The user can examine the actual quadtree or octree node values.

The remaining items are internal tables of interest only during system development or in the examination of

the state of the system after an error has occurred.

KEY 29: FILE - This key allows the user to save an object in a disc file or restore an object from a disc file. The save or restore selection is picked from the message field followed by the keyboard entry of the file name. The object files are kept in a Prime Sub-UFD. The name of the Sub-UFD is specified by a DATA statement in the GP/1 insert file (GP/1.INSERT) for array DFSUB$. It currently specifies the name as 'S_OCTREE_DATA_FILES' but can be easily changed by modifying DFSUB$ in COMMON.

There are currently four data file types defined. The correct file type is determined automatically.

If an object is restored from a disc file, its object number is displayed.

The following function key is used to display 3-D octrees:

KEY 23: VIEW 3D - This key is used to display an octree. A view can be generated of the forward edges on the Imlac 6220 terminal or a shaded surface view on the DeAnza display.

Numerous options can be specified. In general, the options should be selected before the object or objects are selected.

If an octree object is picked (no options specified), a second object, a "tagged" object is first generated. This is a new version of the object in which

F nodes have been given additional information indicating the visibility of their edges. A second object number must be available for each object displayed. The conversion to tagged format is noted in the menu field with a listing of the old object number (still retained as an object) and the corresponding tagged object number. The tagged objects are not deleted at the end of the display operation so that they can be viewed later without requiring conversion. Note: The tagged object number should be selected in later display operations or a new tagged object will be generated. The tagged object cannot be used with operations other than display. They cannot be translated or unioned, for example.

If option '/MULTIPLE/' is selected, multiple objects can be selected for simultaneous display. A new option, '/END/' is presented in the message field. It is selected to terminate object entry and start image generation.

Option '/DEANZA/' requests that the image be generated only on the DeAnza raster display. If '/DZ & IMLAC/' is selected, both a forward edge display (on the Imlac) and a shaded surface display (on the DeAnza) will be generated. The default is to generate a forward edge display only.

If '/UNIV/' is selected, the outline of the entire universe will be displayed as an aid in determining

object location and orientation.

If '/CUBES/' is picked, the conversion to tagged format is inhibited. The object is presented on the Imlac screen as a collection of cubes (corresponding to F nodes). Note: This can quickly overflow the Imalc's vector storage memory and should only be used for simple objects defined at the higher levels.

Option '/MINLEV/' allows the user to specify a minimum tree level for display. At this level, all P nodes are interpreted as F nodes for display.

Option '/B & W/' specifies that only a black and white image is to be generated on the DeAnza. Normally, the first 10 items selected for display are given fixed colors (see key 14). If this option is selected, only the red value is used. It is used as an intensity for white (copied into all three color channels). An advantage to generating a black and white image is an increase in image generation speed. This option, of course, makes no difference for vector display on the Imlac terminal.

The option '/SUBPIXEL/' requests that the image to be displayed on the DeAnza first be generated in computer memory, then averaged to a higher level before transfer to the DeAnza. This is typically done to reduce the aliasing problem.

The '/PERSPCT/' option selects a perspective view

generation. The use of this option is not recommended for general use at this time because only a single tree traversal sequence is used and the resulting image is correct, in general, only for one quadrant of the screen (see octree references). This capability was implemented simply to verify the perspective view generation algorithm.

The user is next asked to enter the center point on the screen for the image, using the lightpen. If the tracking cross is confirmed in place, the image center will be the center of the Imlac screen. If numerous images are to be displayed on the screen, a different location should be indicated for each.

The user next selects the three view angles using the lightpen. Three markers can be moved in three sets of axes in the upper left hand corner of the screen. Simply move the tracking cross to the next desired angle within the axis area. Confirm the tracking cross location to enter the angle. The angles can be changed indefinitely. To terminate angle entry, confirm the tracking cross within the small square where it is initialized (or, in other words, confirm the last point twice). To use the default viewpoint, simply confirm the initial location without movement. Note: At the current time, the viewpoint must be located at a positive displacement in all 3 axes. To view other sides of an

object, it should be rotated by 90 degrees in the appropriate axes.

Next, the scale factor is entered. For a scale factor of 1.0, the universe containing the object or objects would exactally fill the screen if viewed from a location on an axis.

It is recommended that a small scale factor (0.2 to 0.4, say) be used initially to insure that the object will be visible within the view window.

If a negative scale factor is entered, the absolute value is used and the DeAnza screen is cleared.

Three options now become available if a DeAnza display was specified. A subpixel generation at one level below pixel level is the default (each pixel is the average of a 2 by 2 set of subpixel intensity values). Lower levels can be specified at this time. This is not usually recommended because each additional level will typically increase processing time by a factor of four.

The '/NEW WINDOW/' option allows the specification of a new window on the virtual viewplane. A location and a size are specified in x and y. The use of the default parameters is recommended until the user gains some experience in display generation.

The default window is 128 pixels by 128 pixels located at 192,192 (128 pixels centered on a width of 512). This will allow the generation of a full color

image at a subpixel level difference of 1 with a single pass. Larger window sizes are automatically generated in multiple passes. If '/B & W/' is specified, a 256 pixel square window can be generated in one pass (subpixel level of 1). The recommended window origin is 128,128 to center the window (specify '128,128,256,256').

The option '/NEW LOC/' will allow the image to be located anywhere on the DeAnza screen. This is important if multiple images are to be generated. The location of the origin of the window (lower left hand corner) is placed at the specified location on the screen (0 to 512 in x and y with 0,0 the lower left hand corner of the screen).

After image generation, a new view can be generated. The last view is either erased or retained (user selectable).

KEY 14: COLORS - This key allows a new set of color values to be specified for objects to be displayed. The option '/NEW COLOR TABLE/' should be selected (the other options are used to change the shaded surface intensity generation parameters and should not normally be used). After selection, a file name is requested.

The file is an ASCII file in the main UFD (not a Sub-UFD). It should contain 10 lines, each with three integer numbers (from 0 to 255) separated by commas. The three numbers represent the red, green and blue

intensities, respectively.

The individual lines represent the particular objects to be displayed. The first object selected for display (any object number) is given the color set from the first line. The second is from the second line, and so on. Note that the color is not associated with the number of an object but only with the order of object selection.

4 GP/1 - Programming Information

4.1 System Loading

The magnetic tape accompaning this manual contains two logical tapes recorded using the Prime Computer MAGSAV utility at 1600 bpi. The first logical tape contains the GP/1 system and the second contains the OCTREE system. Each contains two files.

On the first logical tape, file 'GP/1_FILES' contains the GP/1 system. The second, 'C_GP/1_INIT' is a command file. When run ('CO C_GP/1_INIT') it will unload the appropriate files, recompile the system and set up the demonstration version of GP/1.

4.2 File Organization

The GP/1_FILES file contains a number of "blocks." A block can be a main program, a subroutine, an insert file, an information file, etc. The first line of a block begins with the three character string 'C//' beginning in column 1. The format of the line is:

C// name number description

The name field contains the name of the block. It typically corresponds to a subroutine name or other program unit name.

If the block is separated from the main file and placed into its own file, this should be the name of the new file. When removed, a line of the form 'C//> name' is placed in the original file to mark the removal location. A subroutine is typically removed from a subroutine library for modification and testing. It can be compiled without recompiling the entire set of subroutines. Its binary file is loaded in the load command file before the binary for the library.

At a later time when a desired set of changes have been accomplished and tested, the removed files are reinserted and the set of subroutines recompiled as a unit. The individual files and their binary files are then deleted and the load commands removed from the load command file.

In the Prime system a particular block can easily be removed using the source editor with the following set of commands:

```
F C// name
DUN name TO C//
N-1
I C//> name
```

It is later placed back into the original file with the following commands:

```
F C//> name
D
LOAD name
```

Every block is given an identification number. Such

numbers have been reserved in blocks as follows:

1 to 150 - GP/1 System

151 to 200 - Applications Library

201 to 700 - Program OCTREE

701 and up - other applications

Within a block, numbers are generally allocated sequentially as new modules are developed. When a new version of an existing program is developed or in the case of support subroutines, a fractional part is added to the number of another block. A new version of subroutine 140 could become 140.1, for example.

At appropriate times during system development, all files are integrated into a single file in numerical order. A listing is generated with line numbers. For an index, a COMO file is initialized and the editor given the following commands:

```
MODE NUM
F C//;*
```

A list of the line numbers and headers is thus generated. The Prime sort program can be used to place the index in alphabetical order.

4.3 Compile Options

All modules are compiled using the Prime FTN compiler with options '-BIG' and '-INTL'. All integers are *4 (4 bytes) unless specified otherwise.

4.4 GP/1 Structure

The structure of GP/1 is shown in Figure 4.1. A very small main program is used. In Figure 4.1 it is called GP/1.MAIN but in practice is typically named according to the application. The main program is usually kept as a separate file during development for ease of modification and compilation. It must be the first routine loaded.

The main program is essentially identical for all applications. It calls one of 32 subroutines (actually only 30 are available because of 'HD COPY' and 'EXIT') depending on the function key selected by the user. The subroutines called by the main program are all named 'FUNn' where n is an integer function number. The function number is unrelated to function key number (0 to 31) or block number. They can be moved freely to any key (except 15 and 31) as desired by the programmer.

Function numbers 1 to 200 are reserved for GP/1. Numbers 201 and above are available for applications.

The function subroutines, both GP/1 functions and

STRUCTURE OF GP/1

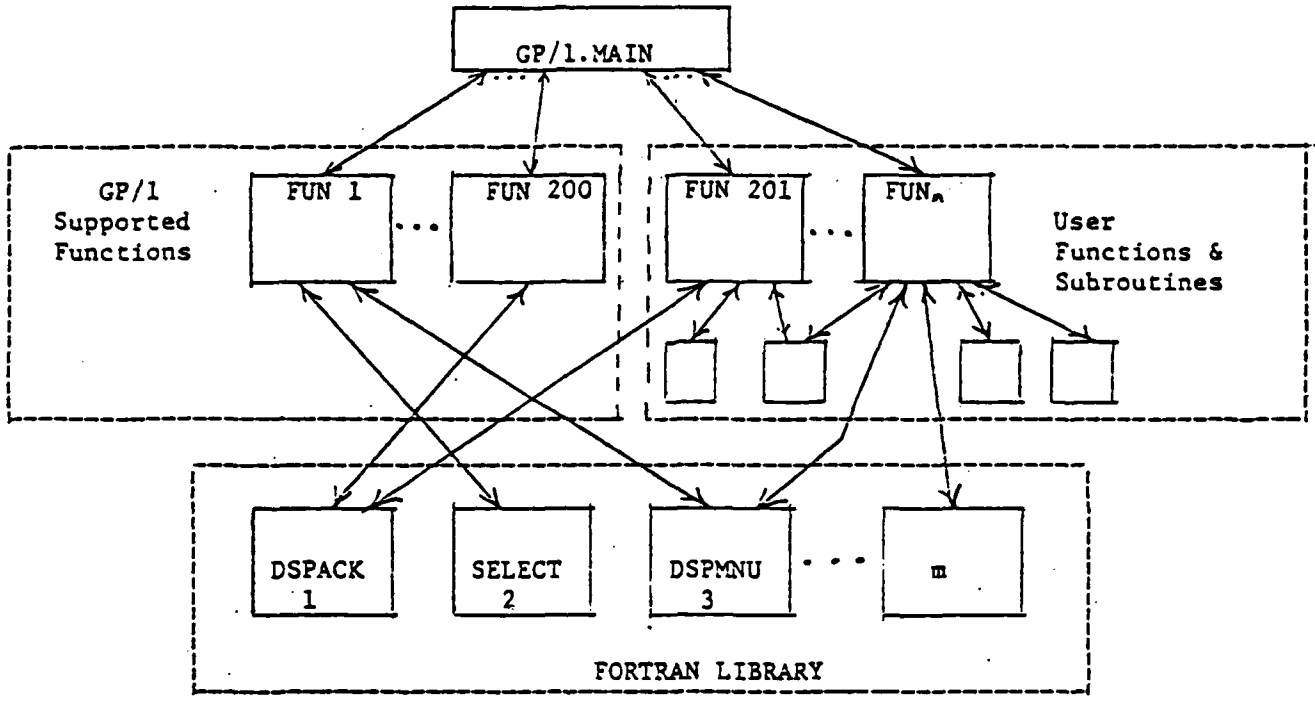


Figure 4.1 GP/1 Subroutine Library (GP/1_FILES)

application subroutines can draw upon the GP/1 subroutine library to perform GP/1 operations (post a message, enter a point into the tables, etc.). User functions typically draw upon a set of application related subroutines also. There is no naming or numbering convention for these lower level routines.

The main program is configured as shown in Figure 4.2. A data statement defines the 8 character labels for the function keys. If not used, they should be set to all blanks.

Subroutines are next called to initialize system tables and the Imlac screen.

All lightpen picks are handled by a single subroutine called SELECT. It returns the subpicture type in variable ITYPE. This value is reduced by the base number for the block of subpicture-type numbers reserved for function keys (variable 'FUNKEY') to indicate the key number selected. If a valid key number in not generated (user selected other than a function key), the selection is rejected (the 'GO TO 99'). Otherwise, a computed GO TO is used to jump to the continue statement with a statement number corresponding to the key selected (1 to 31 with key 0 falling through). The continue statements are followed by a call to the appropriate function subroutine. A 'GO TO 99' is executed immediately upon return from the function subroutine so as to request the next key selection.

```
      DATA/KEYS/ (key lables)
      (initialization)
99    CONTINUE
      (clear function key field)
      CALL SELECT(10,NSUB,ITYPE)
      KEY=ITYPE-FUNKEY
      (clear screen fields)
      IF((KEY.LT.0).OR.(KEY.GT.31)) GO TO 99
      GO TO (1,2,3, ... ,32), KEY
      (code for key 0)
      GO TO 99
1     CONTINUE
      CALL FUNn
      GO TO 99
2     CONTINUE
      CALL FUNm
      GO TO 99
      ...
31    CONTINUE
      CALL FUNp
      GO TO 99
1000  CONTINUE
      END
```

Figure 4.2 - Organization of GP/1 Main Program

The main program for the GP/1_DEMO program is contained in Appendix C.

4.5 Loading

The convention for linking and loading a program is to employ a command file named 'C_name' where name is the name of the program ('GP/1_DEMO', for example). The system libraries currently in use are as follows:

HGP1MVL - Imlac Host Graphics Package

VAPPLB - Prime Applications Library

BITLIBV - Graphics Tablet Library

T$DIP - DeAnza Display Library

IP$LIB - Image Processing Library

The GP/1 subroutines are contained in file 'GP/1_FILES'. When compiled, file B_GP/1_FILES is generated. Command file C_GP/1_LIB is used to convert B_GP/1_FILES into a library module called LIB_GP/1_FILES. B_GP/1_FILES is automatically deleted to free up disc space. Within the load command file, the load request 'LI *>LIB_GP/1_FILES' should be specified. It may need to be loaded twice in order to resolve all references.

The use of a library load rather than loading the binary file has the advantage of causing only the subroutines actually used to be placed into the load module, rather than all subroutines.

4.6 Insert File

System definitions, tables, common blocks, etc. are contained in file 'GP/1.INSERT'. It is inserted into every subroutine for communications. The programmer should become familiar with the variables defined in GP/1.INSERT in order to understand their use and to avoid naming conflicts within new programs. A copy of the current GP/1.INSERT is contained in Appendix D.

The request for file insertion should be placed after type and common definitions but before any DATA statements. On the Prime system the following statement is used: '$INSERT GP/1.INSERT' (beginning in column 1).

4.7 Programming Considerations

The programmer should keep a few points in mind. First, the main program places the terminal in graphics mode. This is the normal state of the system. All GP/1 subroutines assume that the program will be in graphics mode when called. If a new routine exits graphics mode (for example, to perform ASCII communications with the user through the keyboard) graphics mode should be reentered before any other subroutine is called.

Second, all subpictures are displayed at the beginning of the main program (during initialization). All are

centered at 0.0,0.0. No subroutine should call for the display of a subpicture.

4.8 GP/1 Data Structure

A number of tables in GP/1.INSERT are used to maintain the state of the program and store information. The subpicture table (ISUBTB) is shown in Figure 4.3. The subpicture number is the primary key used to identify items on the screen. This includes functions, messages, etc. The maximum number of subpictures is contained in variable MAXPIC (currently 128). Array ISUBTB contains two columns. The type field for a subpicture entry indicates the current use for the subpicture. It may, for example, be a point or a line, or could be used as a menu item. A null entry ('EMPTY') indicates that a particular subpicture is not currently being used and is, therefore, available.

The second field, the entry field, indicates the particular item number of the specified type. It typically points to an entry in a table for that item type.

In Figure 4.4, the point storage table is shown. The entry number from the subpicture table indicates the particular record. Three fields are used. The header can be any non-zero integer (0 indicates a null entry) selected by the programmer for a particular application. The two remaining fields contain the x and y coordinate values in

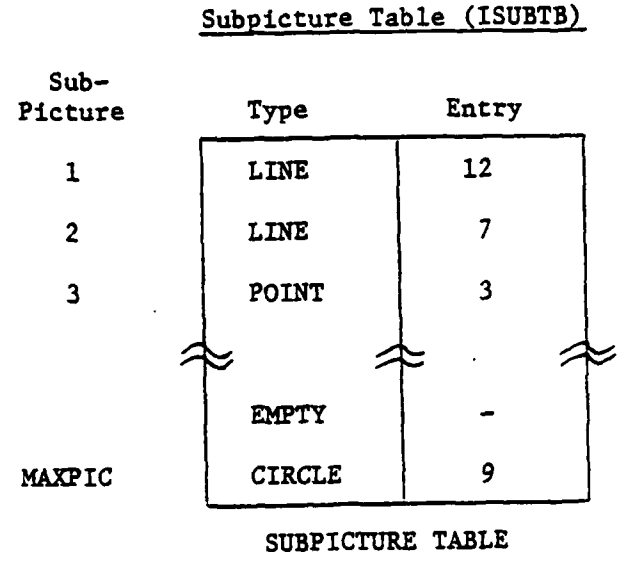


Figure 4.3 Subpicture Table (ISUBTB)

Points (IPTABL & PTABLE)

Entry	Header	X	Y
1	EMPTY	-	-
2	1	2.40	-1.69
3	≈	≈	≈
	1	3.60	0.00
MAXPNT	EMPTY	-	-

POINT TABLE

Figure 4.4 Point Storage Tables (IPTABL and PTABLE)

floating-point format. The two actual tables forming the point table are IPTABL (integer part) and PTABLE (floating-point part).

The line table and circle/arc tables are shown in Figure 4.5. They are defined in a manner similar to that for the point table.

Figure 4.6 shows the manner in which the various tables are used to maintain items on the screen. The entry in the subpicture table corrseponds to the screen subpicture used to display the item. To loacte the screen subpicture for a particular point, for example, the subpicture table is searched for a point with an entry number corresponding to the desired point. This identifies the subpicture. It could now be erased from the screen, be made to blink, etc. In the other direction, a lightpen pick on a particular subpicture uniquely identifies an item through the tables.

The steps required to enter a point into the tables (a "PUT" operation) are shown in Figure 4.7. Figure 4.8 lists the steps to perform a "GET" operation (read point values based on subpicture number) while Figure 4.9 shows the procedure to delete a point from the system based on subpicture number (a "DEL" operation).

The storage of chains is shown in Figure 4.10. The "chain table" (ICHTB) contains a single entry of six fields for each chain. The actual link values for all chains are contained in a single vector, the "chain value table"

Lines (ILINTB & FLINIB)

Entry	Header	VI	V2	V3	V4	V5
1	1	1.20	-1.55	3.10	3.15	0.00
2	EMPTY	-	-	-	-	-
3	EMPTY	-	-	-	-	-
	≈	≈	≈	≈	≈	≈
MAXLIN	1	4.21	1.30	4.32	1.00	0.00

LINE TABLE

Notes: (1) Currently, values are defines as:
V1 - X value of point 1
V2 - Y value of point 1
V3 - X value of point 2
V4 - Y value of point 2
V5 - (not used)

Circle/Arc (ICIRTB & CIRTB)

	Header	Center X	Center Y	Radius	Angles (Radians) Start	Angles (Radians) End
1	EMPTY	-	-	-	-	-
2	1	2.51	3.20	0.91	0.00	6.28
3	1	1.91	-3.21	1.00	0.91	0.98
	≈	≈	≈	≈	≈	≈
MAXCIR	EMPTY	-	-	-	-	-

CIRCLE/ARC TABLE

Notes: (1) Arc is counterclockwise from START angle to END angle.

Figure 4.5 Line and Circle/Arc Tables

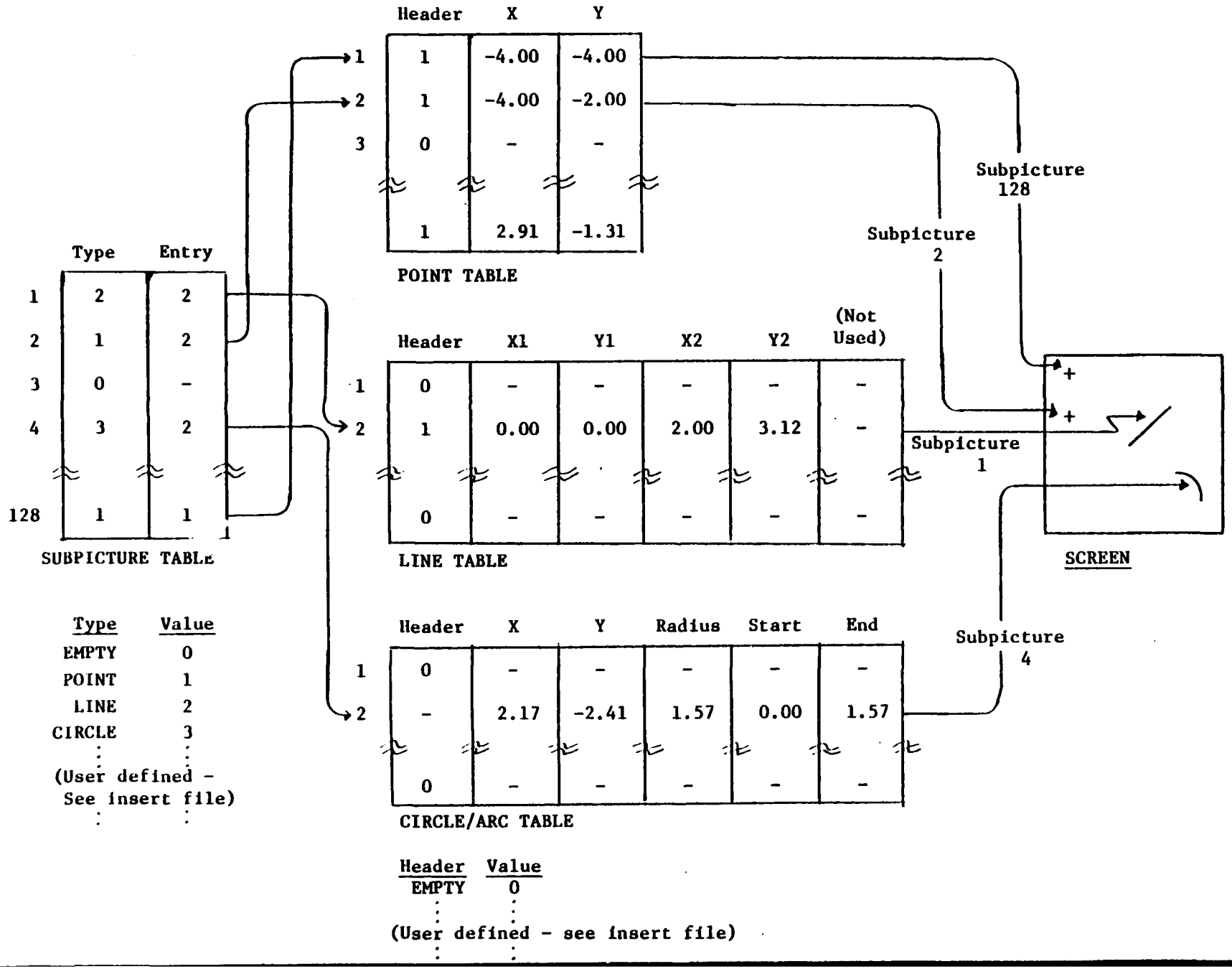


Figure 4.6 GP/1 Subpicture Example

EXAMPLE

PUT Operation: ENTER POINT AT (1.00,1.00)

1. Search for empty subpicture in SUBPICTURE TABLE (error if none available).

2. Enter POINT in TYPE field of SUBPICTURE TABLE.

3. Search for empty entry in POINT TABLE (error if none available).

4. Enter header value (not 0!) in HEADER field of POINT TABLE. Enter location values in X and Y fields of POINT TABLE.

5. Put the entry number from POINT TABLE into ENTRY field of SUBPICTURE TABLE.

Figure 4.7 Steps in a "PUT" Operation

GET Operation: Obtain end points of line in subpicture 1

1. Read TYPE field of entry 1 of SUBPICTURE TABLE. If not equal to LINE (Value of 2), error.

2. Read ENTRY field of entry 1 of SUBPICTURE TABLE.

3. Read HEADER value of LINE TABLE for entry from (2). If header value is equal to EMPTY (value of 0), error.

4. Read data fields from LINE TABLE and return.

Figure 4.8 Steps in a "GET" Operation

DEL (Delete) Operation: Delete arc in subpicture 4

1. Read TYPE field of entry 4 of SUBPICTURE TABLE. If not equal to CIRCLE (value of 3), error.

2. Read ENTRY field of entry 4 of SUBPICTURE TABLE.

3. Read HEADER value of CIRCLE/ARC TABLE for entry from (2). If value equal to EMPTY (value of 0), error.

4. Set HEADER field of CIRCLE/ARC TABLE to EMPTY.

5. Set TYPE field of SUBPICTURE TABLE to EMPTY (value of 0).

Figure 4.9 Steps in a "DEL" Operation

IMPLEMENTATION OF NEW ITEM TYPE: CHAINS

- Check GP/1.INSERT for next available TYPE number.
- Set up CHAIN TABLE:

	Header	Level	Start in Value Table	No. of Elements	Root of Chain X	Root of Chain Y
1	EMPTY	-	-	-	-	-
2	1	5	420	131	412	919
≈	≈	≈	≈	≈	≈	≈
MAXCH	1	2	1	418	43	937

CHAIN TABLE

Set up CHAIN VALUE TABLE

1	1	First Value - CHAIN 2
2	1	
3	2	
≈	≈	
418	7	Last Value - CHAIN 2
419	-1	End Marker
420	3	First Value - CHAIN MAXCH
≈	≈	
550	4	Last Value - CHAIN MAXCH
551	-1	End Marker
552	-2	End of Data
≈	≈	
MAXCHV	-2	

CHAIN VALUE TABLE

Figure 4.10 Chain Storage

(ICHVTB). The starting location in ICHVTB and the number of links are stored in the chain table. New chain link values are added to the end of the table. When a chain is deleted, lower values in the table are moved up to fill the gap. The chain table must, of course, be updated.

The most commonly used GP/1 subroutines are presented in Appendix E along with examples. A detailed tabulation of all current GP/1 subroutines is presented in Appendix F (including argument definitions).

The subroutines in the current applications library are presented in Appendix G.

5 OCTREE - Programming Information

5.1 System Loading

The second logical tape on the associated magnetic tape contains two files, 'OCTREE_FILES' and 'C_OCTREE_INIT'. It is assumed that the GP/1 files have been loaded and the command file ('C_GP/1_INIT') run in the same UFD. The command file 'C_OCTREE_INIT' should now be run ('CO C_OCTREE_INIT'). The appropriate files will be unloaded, compiled and formed into the run module, '#OCTREE'.

5.2 Organization

The OCTREE files are organized using the same conventions as those used for the GP/1 files.

The OCTREE system is built upon and integrated into the GP/1 system. The new main program is 'OCTREE'. A copy is contained in Appendix H. The load command file is 'C_OCTREE'.

When 'C_OCTREE_INIT' is run, the system-wide insert file is loaded back into 'GP/1_FILES' and a new insert file taken from 'OCTREE_FILES'. This is an updated insert file containing additional data structures. Appendix I presents the new insert file.

The following new insert files are also used:

COM254 - Communications rooted in Subroutine 254

COM257A - Communications rooted in Subroutine 257 (before GP/1.INSERT)

COM257B - Communications rooted in Subroutine 257 (after GP/1.INSERT)

DIPDEF - Keys and Unibus addresses for DeAnza display

5.3 Loading

Loading is identical to that for GP/1 with the addition of the new library 'LIB_OCTREE_FILES'.

5.4 The OCTREE Data Structure

Quadtrees and octrees are stored in the same data structure. It is organized in a structure similar to that used for chains. A small two-dimensional array contains information concerning the objects while a long one-dimensional array contains the node values.

The information array is 'IARTB' for "area table" (during the initial development of this program, quadtrees were called "area encoded" objects). The node values are in 'IARVTB' (for "area value table").

IARTB is a 2-D array, the first parameter specifying the

object number. The second argument selects the following information items (all 32-bit integers):

HEADR$ - Header value (see definitions below).

LEVEL$ - Lowest level (0 to 10).

START$ - Starting location in IARVTB.

NOELE$ - Number of words (16-bits) in IARVTB used to store object nodes (usually is also the number of nodes).

ORGNX$,ORGNY$,ORGNZ$ - Level 0 value of origin of local coordinate system.

SCALX$,SCALY$,SCALZ$ _ Scale factors of object.

USER1$,USER2$,USER3$,USER4$ - Values available for use by application programs.

The current formats for origin and scale values can be found in the insert file ('GP/1.INSERT').

The bits in the header word are allocated as follows (the LSB is bit 0):

Bits 0 to 2: Dimensions - Number of dimensions object defined over (1 to 7).

Bit 3: Tag Bit - Tagged object if set.

Bit 4: Tag Type - Raster tagged object if set, vector if reset.

Bit 5: Test Object - Test object if set (used in display program).

Bit 6: Display Inhibit - Object not to be displayed if

set. (Typically used as non-display interference object.)

Bit 7: Interference Check - If set, interference with this object should be checked during display.

Bit 8: Transparent Object - This object is transparent if set.

Bit 9: Add/Subtract - If set, this object has a negative visual density.

Bit 10: GS Object - This object is a GS object if set.

Bit 11: SN Object - This object is an SN object if set.

In general, the OCTREE subroutines operate independently. They typically operate on one or two objects contained in the IARTB table, often generating another object. Most perform a single function or series of functions on the objects.

Subroutines leave system tables (including object tables) in a standard configuration when they terminate (if no errors occurred). They can thus, in general, be called randomly by the user or by higher-level routines with full confidence that system integrity will be maintained. Extensive error checking is performed before and during execution of object operation.

The main program for program OCTREE is presented in Appendix H. The new version of the system insert file

(GP/1.INSERT) is presented in Appendix I.

The OCTREE subroutines that are generally available are listed in Appendix J. The theoretical foundations embodied in the routines are presented in the octree references listed in Chapter 1. Details of the implementation of individual algorithms are documented within the source code. They are not repeated here for brevity. In general, a programmer should be able to easily make use of the subroutines in Appendix J without actually understanding their internal operation.

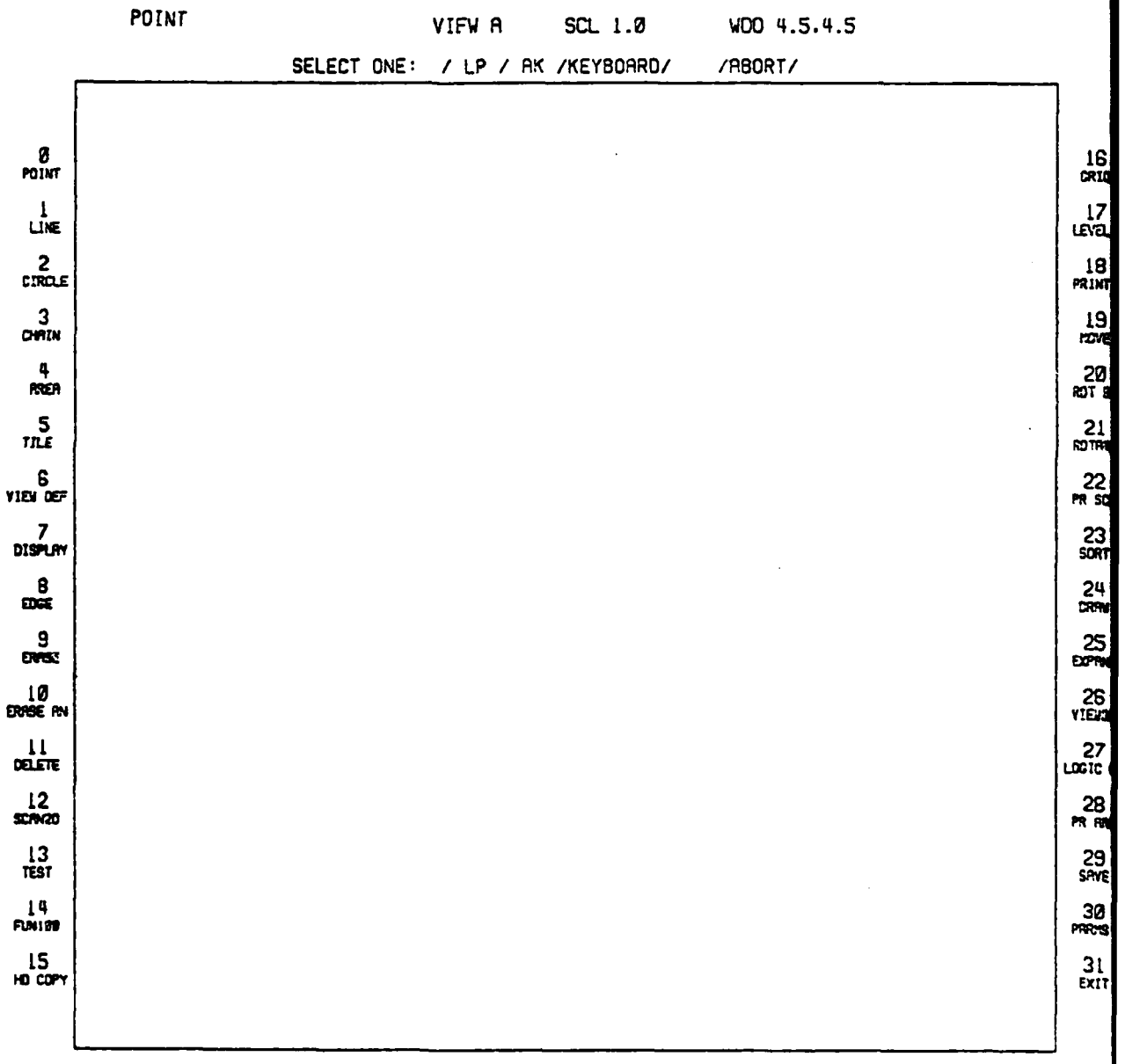
APPENDIX A

EXISTING GP/1 KEY DEFINITIONS

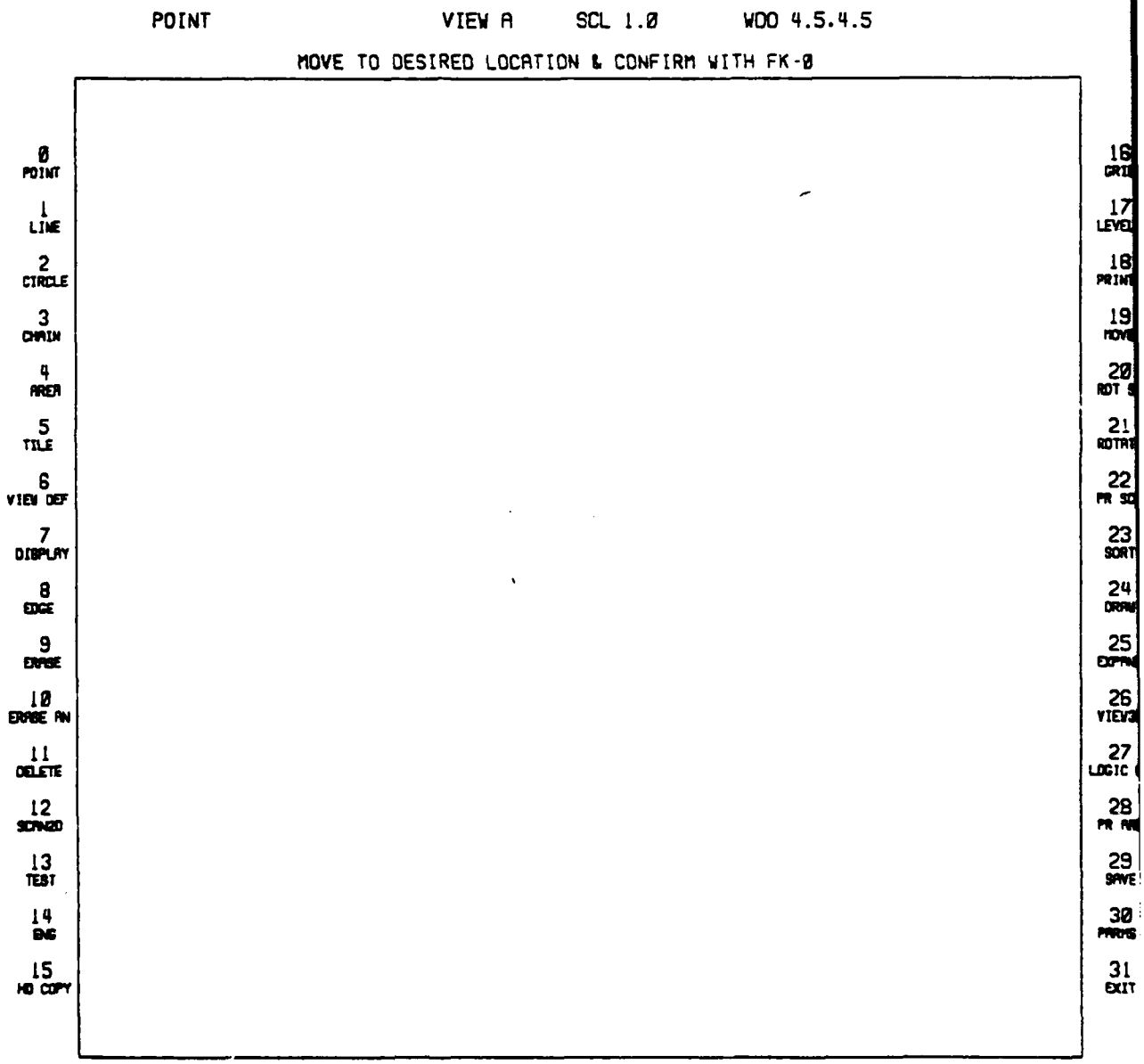
KEY 0 - POINT

A point is entered and displayed in a new subpicture. An entry format must first be selected from the MESSAGE field:

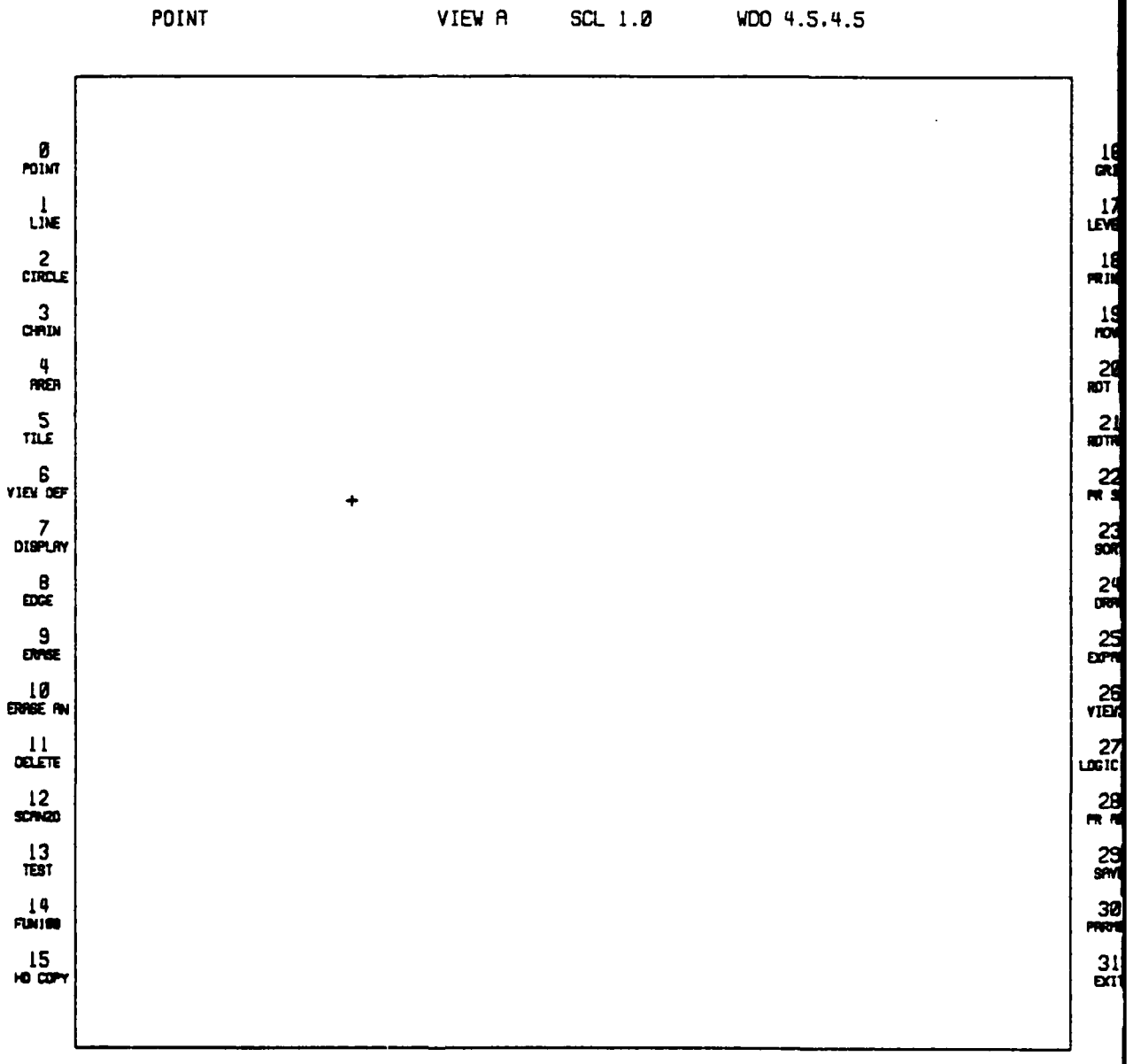
LP - Lightpen input
AK - Arrow key input
KEYBOARD - Keyboard input of X,Y in standard units (-4.5 to 4.5)



POINT ENTRY WITH LIGHTPEN (1/3)



POINT ENTRY WITH LIGHTPEN (2/3)



POINT ENTRY WITH LIGHTPEN (3/3)

KEY 1 - LINE

A line is entered and displayed in a new subpicture. An entry format must first be selected from the MESSAGE field:

BOTH POINTS: KEY - Enter both points from keyboard (X1,Y1,X2,Y2) in standard units (-4.5 to 4.5)

BOTH POINTS: LP - Enter both ends of line with lightpen

BOTH POINTS: POINTS - Enter both ends of line by picking two existing (displayed) points with lightpen

FIRST POINT: KEY - Enter first point from keyboard (-4.5 to 4.5)

FIRST POINT: LP - Enter first point with lightpen

FIRST POINT: POINT - Enter first point by picking existing (displayed) point with lightpen.

If a "FIRST POINT" option was selected, the user will be presented with the same three choices for entry of the second point.

KEY 2 - CIRCLE

A circle or arc is entered and displayed in a new subpicture. Prompts appear in the MESSAGE field.

LOCATE CENTER WITH LP (CONFIRM WITH FK-0)

The user is prompted to enter the center of the circle.

KEY: RADIUS (INCHES), ST ANG (DEG), END ANG (DEG)

The user is prompted to enter the radius of the circle (standard units), the start angle and the end angle (both in degrees). An arc is drawn from the start angle in a counterclockwise direction to the end angle.

KEY 5 - TILE

This function (FUN101) demonstrates subroutine TILE (Subroutine 83). The following options are displayed in the MESSAGE field:

TRIANGLE -

The user enters the three points with the lightpen which define a triangle (confirm with FK-o). Note: All triangles will tile the plane.

QUADRILATERAL -

The user enters the four points with the lightpen which define a quadrilateral (four sided shape). Confirm points with FK-o.

KEYBOARD ENTRY -

The user first enters the shape code of the object to be entered:

0 - triangle

1 - quadrilateral

2 - hexagon - type 1

3 - hexagon - type 2

4 - hexagon - type 3

The user next enters the appropriate number of X, Y pairs to define the corners of the object.

EXAMPLES - A second menu is presented in the MESSAGE field for predefined shapes (one of each). They are as follows:

TRIANGLE - sample triangle

QUADRILATERAL - sample quadrilateral

HEX 1 - sample hexagon type 1

HEX 2 - sample hexagon type 2

HEX 3 - sample hexagon type 3

After entry, the shape is displayed.

To continue, the user picks/CONTINUE/ in the MESSAGE field with the lightpen. A normalized copy of the repeating pattern is presented in the center of the screen.

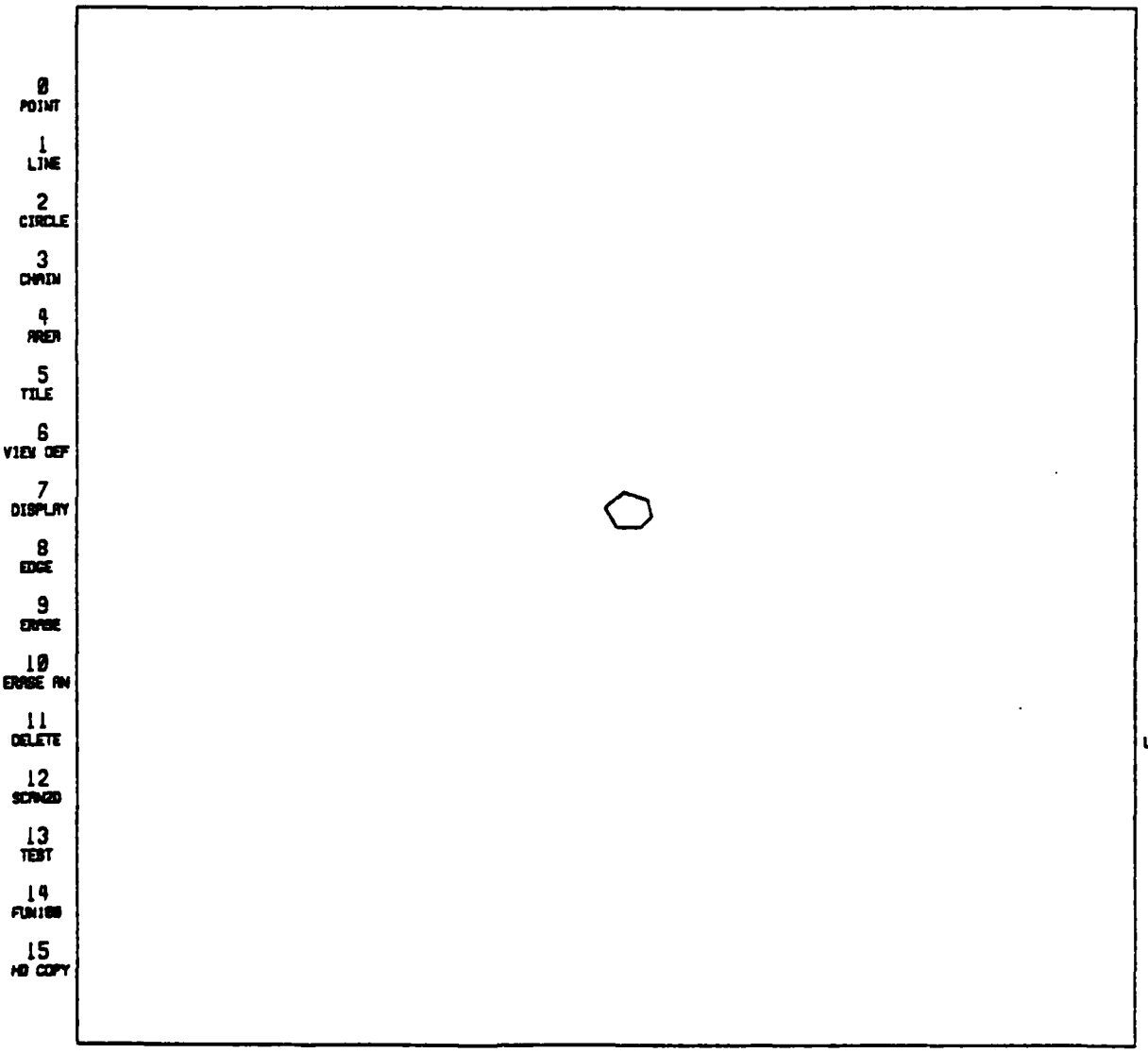
To continue, the user picks/CONTINUE/ in the MESSAGE field with the lightpen. Nine copies of the repeating shape are presented in a 3 by 3 pattern.

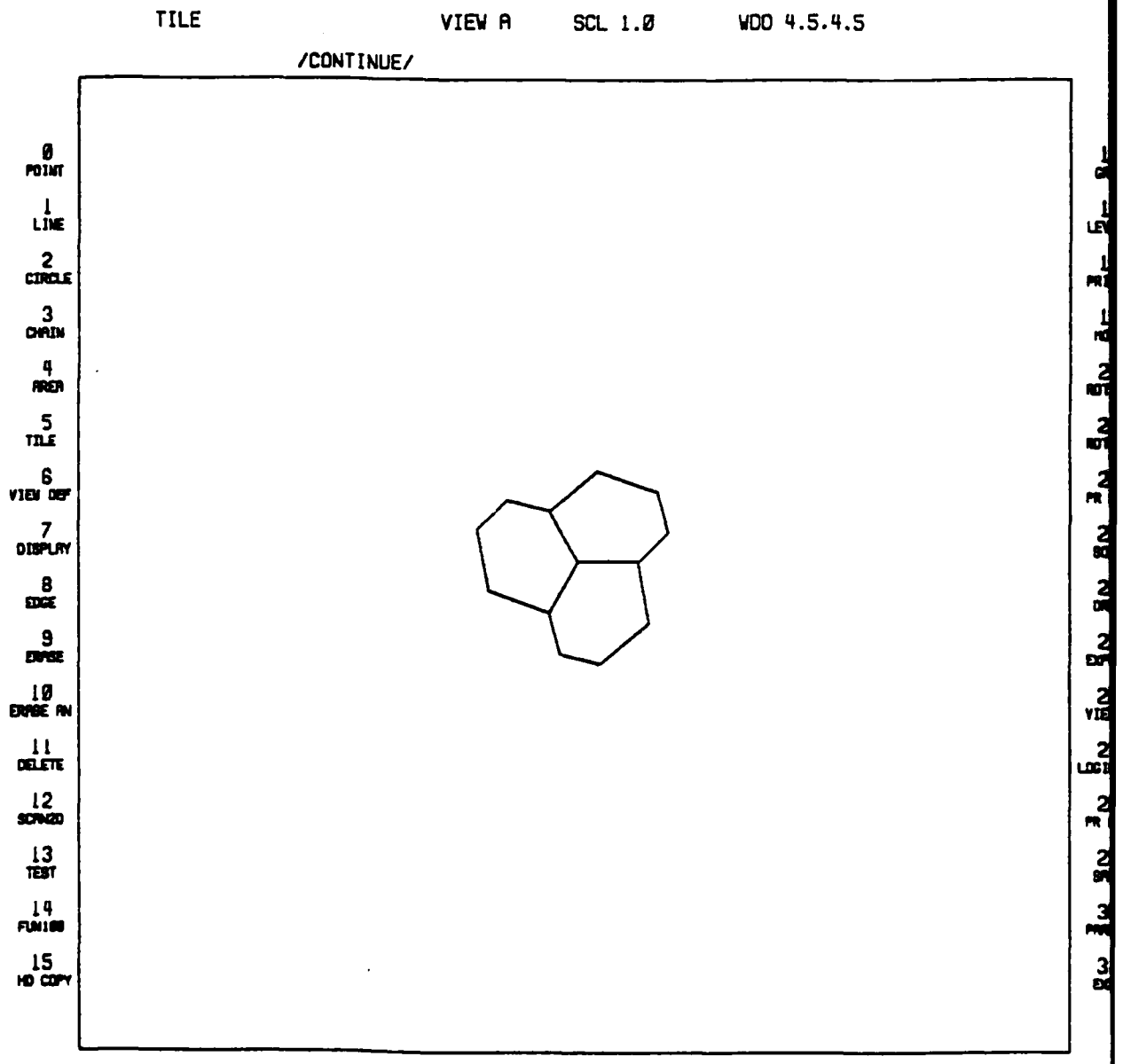
NOTES: (1) The final presentation can be erased with the ERASE key (KEY 9).

(2) For hexagons, the program does not check for compliance with the type rules. If there is a violation, gaps or overlap will occur.

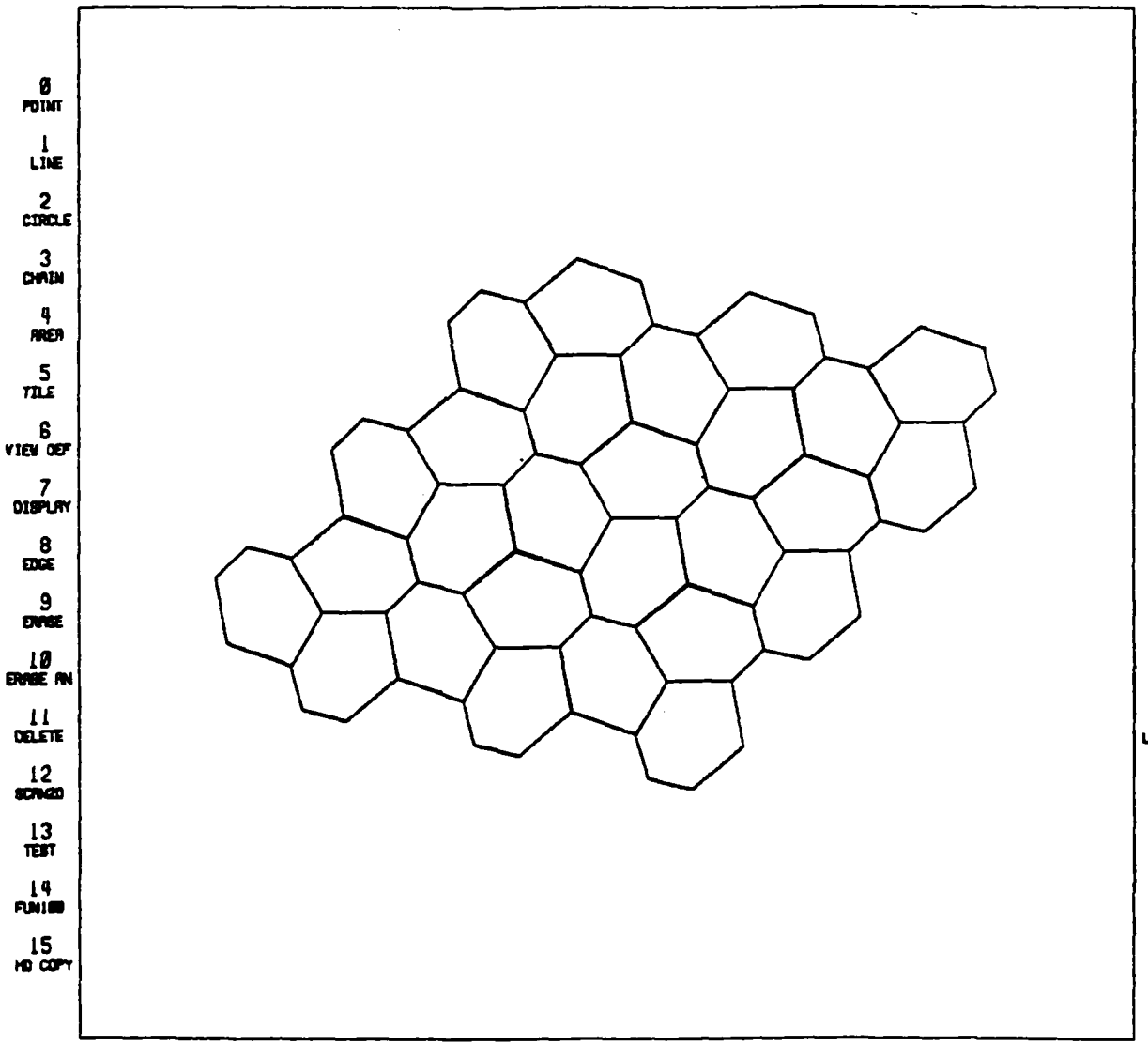
TILE VIEW A SCL 1.0 WDO 4.5.4.5

/CONTINUE/




TILE
VIEW A
SCL 1.0
WDO 4.5.4.5
/CONTINUE/
0 POINT
1 LINE
2 CIRCLE
3 CHAIN
4 AREA
5 TILE
6 VIEW DEF
7 DISPLAY
8 EDGE
9 ERASE
10 ERASE AN
11 DELETE
12 SCRNZO
13 TEST
14 FUNIOO
15 HD COPY

TILE VIEW A SCL 1.0 WDO 4.5.4.5



KEY 6 - VIEW DEF (Viewspace Definition)

This key allows the user to change any of the viewspace parameters. This key is not associated with any viewspace subpictures; it simply allows the changing of the values. Prompts appear in the MESSAGE field.

Note: All values are saved. A single parameter can be changed without resetting other parameters.

SELECT VIEWSPACE/MAIN/1/2/3/4/5/6/7/8/

The user selects the viewspace to be modified.

SELECT FUNCTION/RESET/WIND CLIP/SCALE/TRANS/ROT/VIEW CLIP/STAT/CLOSE/

RESET - Reset viewspace to default conditions and close.
WIND CLIP - Window clipping. The user is prompted to enter XMIN,XMAX,YMIN,YMAX on the keyboard.
SCALE - Scale factor. The user enters the X and Y scale factors on the keyboard.
TRANS - Translation. The user enters the X and Y translation values on the keyboard.
ROT - Rotation angle. The user enters the rotation angle in degrees on the keyboard.
VIEW CLIP - Viewport clipping. The user is prompted to enter the lower left hand corner and upper right hand corner with the lightpen.
STAT - Set status. The user can select the status (ON or OFF) of the following transformations with the lightpen (default is OFF)
WINDOW CLIP - Window clipping
GEOM - Geometric transformations
VP CLIP - Viewport clipping
CLOSE - Close the viewspace. This terminates changes to the viewspace definitions.

Notes:
(1) Except for RESET, parameters can be changed until CLOSE is selected.
(2) Changing a geometric transformation or clipping parameter does not automatically enable it. It will be enabled only after "STAT" is selected to modify the status of the operation.

KEY 7 - DISPLAY

The DISPLAY key is used to display screen items which are not currently visible on the screen. Currently programmed to display chains (see below).

KEY 9 - ERASE

The ERASE key is used to erase an item from the screen but not delete it from the data tables. It is selected with a LP pick.

KEY 10 - ERASE AN

All characters which were displayed in alphanumeric mode are erased.

KEY 11 - DELETE

Screen items selected with the lightpen are erased from the screen and deleted from the system data tables.

KEY 15 - HD COPY (Hard Copy)

This key can be picked whenever the lightpen is active (even inside other functions). It will cause a hardcopy of the screen to be generated.

Notes:

(1) Items placed on the screen in alphanumeric mode will not appear in the hardcopy.

(2) A "HARDCOPY ACTIVE" notice will appear in the upper left hand corner of the screen while the hardcopy is being generated. The user should wait until this message is removed before attempting to enter additional requests.

KEY 30 - PARMS (Parameters)

When selected, certain system operating parameters can be changed.

KEY 31 - EXIT

This key will clear the screen and terminate the program.

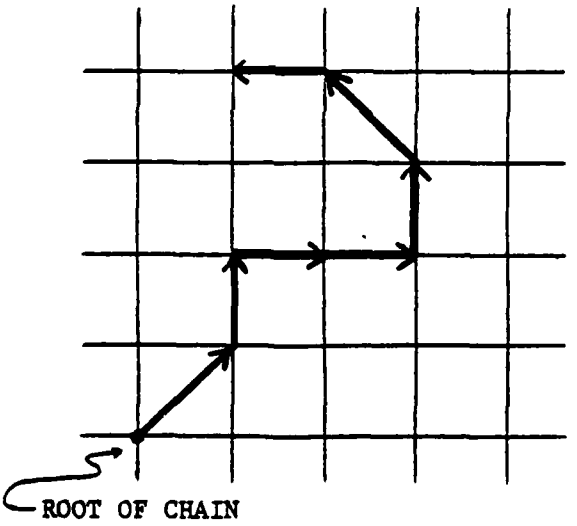
APPLICATION SUPPORT EXAMPLE - CHAIN CODES

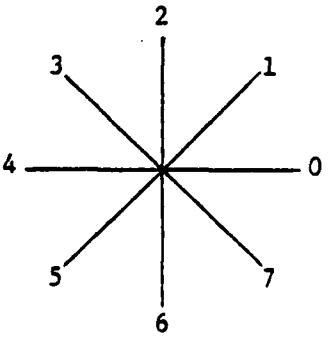
Some of the chain code facilities of an application which used GP/1 have been included in the version generated for this course.

CHAIN CODE

The chain code (Freeman, 1961) is a scheme for encoding arbitrary geometric curves. Based on a rectangular grid, each element of the chain code is a vector connecting a grid point representing a point on the curve to the next grid point (usually and in this case, of the surrounding 8) which most closely matches the curve. The end points of the curve are thus connected by a series of short links. Each vector has a length of 1 or $\sqrt{2}$ times the grid size.

The chain is encoded by a string of octal numbers where 0 represents a vector to the right most neighboring grid point and the remaining seven numbers are generated by stepping in 45° increments in a counterclockwise direction.





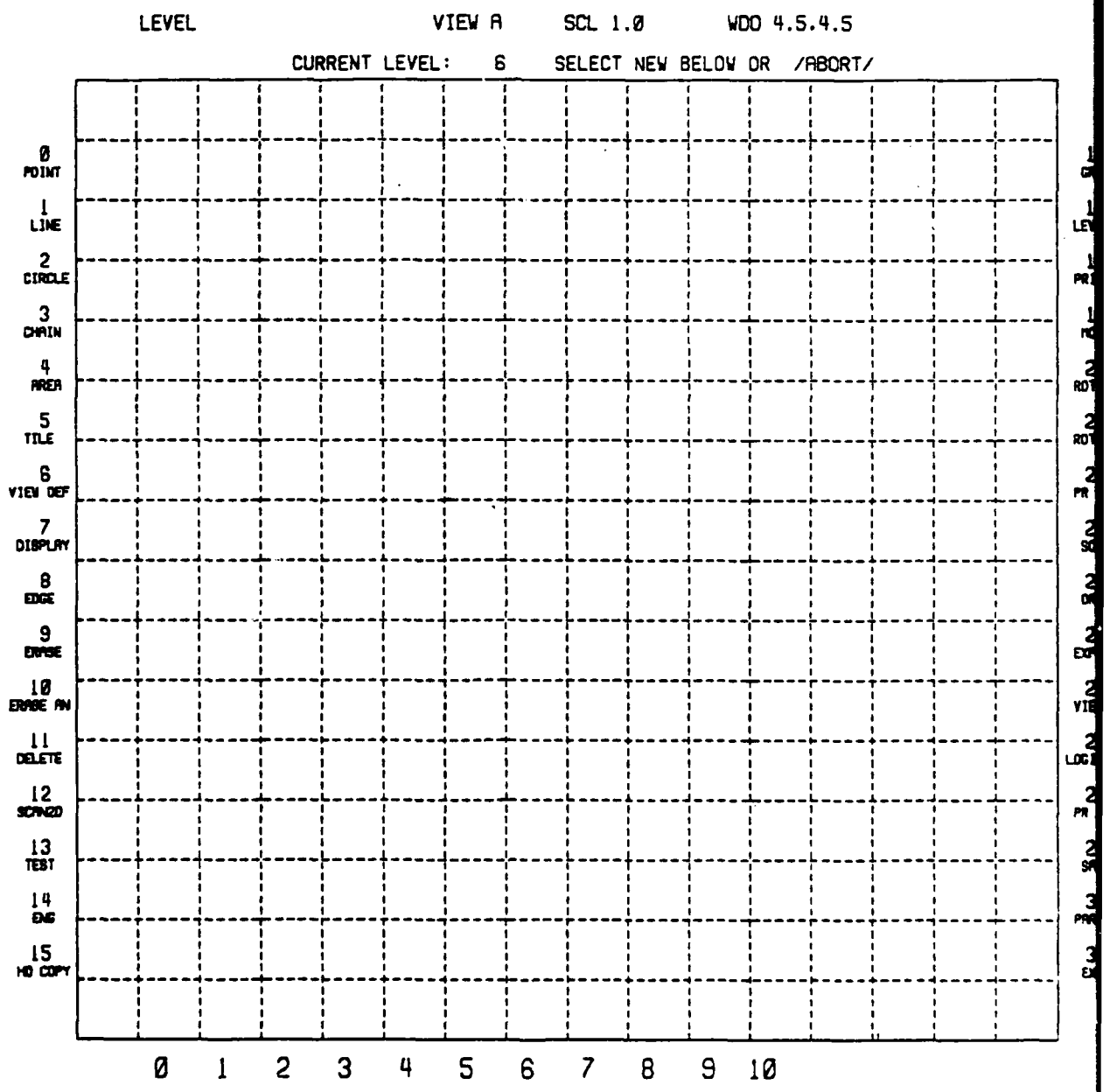
CHAIN CODE

KEY 17 - LEVEL

The level of a chain code refers to the grid size. The lowest level is level 0 in which the working area of the screen is divided into a 1024 by 1024 grid. The number of level 0 lines per grid line at level N is 2**N. Thus, at level 10, the working area repersents a single grid square.

Level	Lines/Grid
10	1024
9	512
8	256
7	128
6	64
5	32
4	16
3	8
2	4
1	2
0	1

When KEY 17 is picked, the current working level is presented and a new level can be selected from the MENU field.



CHAIN - CHAIN GRID LEVEL

KEY 16 - GRID

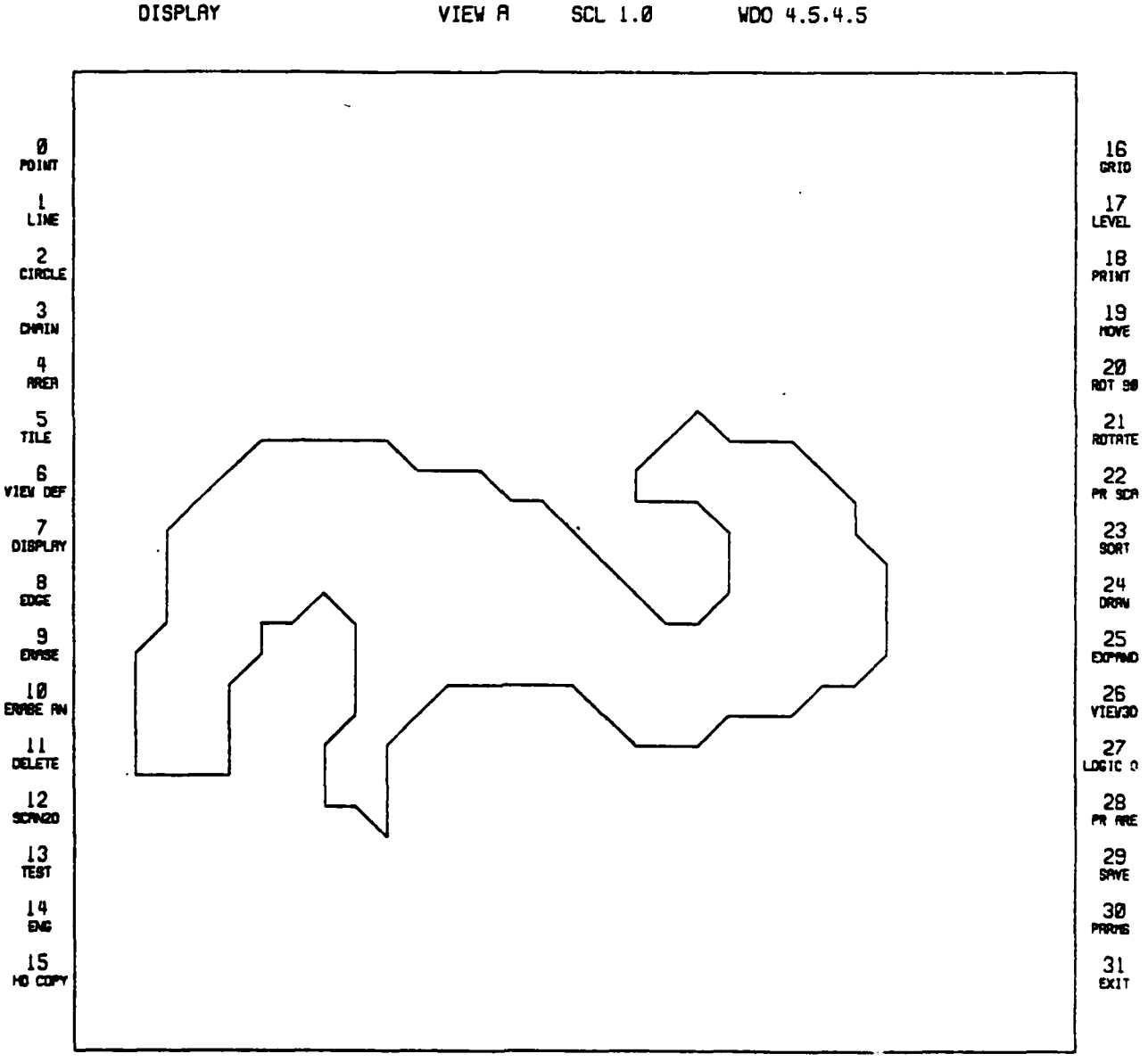
When picked, this key will cause a 32 by 32 line grid at the current level starting at the bottom left of the screen to appear. The next time KEY 16 is picked, the grid is removed.

KEY 3 - CHAIN

This will enable the lightpen tracking cross. It should be moved to a point on the curve and confirmed with FK-0. A chain will now be generated at the current level in response to the movement of the tracking cross. It should be terminated by FK-0.

This function will always generate a closed curve. A shortest length path to the starting point will automatically be generated.

It is not necessary to enable or disable the grid to generate a chain. It will be generated at the current level.



CHAIN AT LEVEL 5 (WITHOUT GRID)

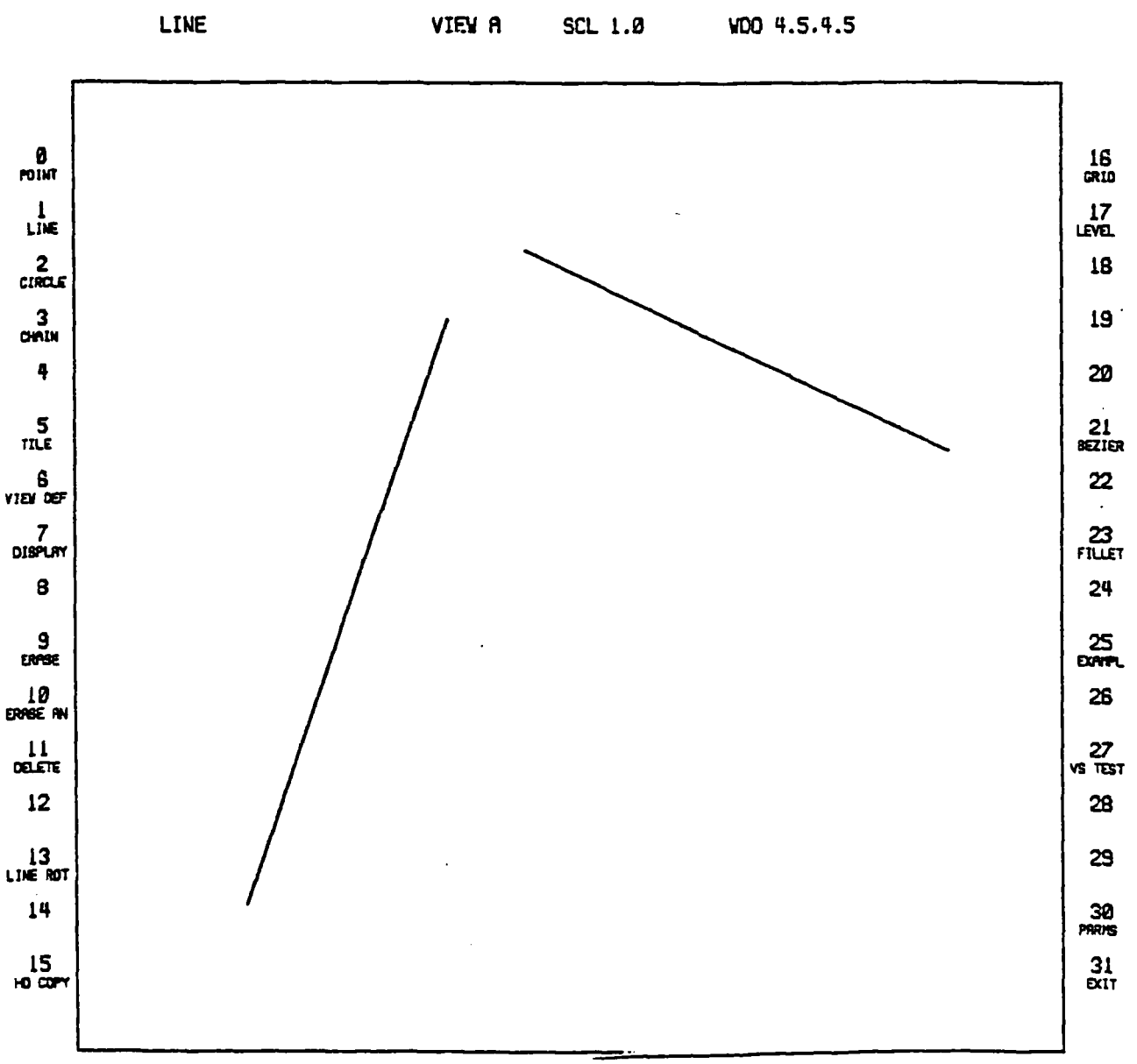
APPLICATION SUPPORT EXAMPLE - FILLET

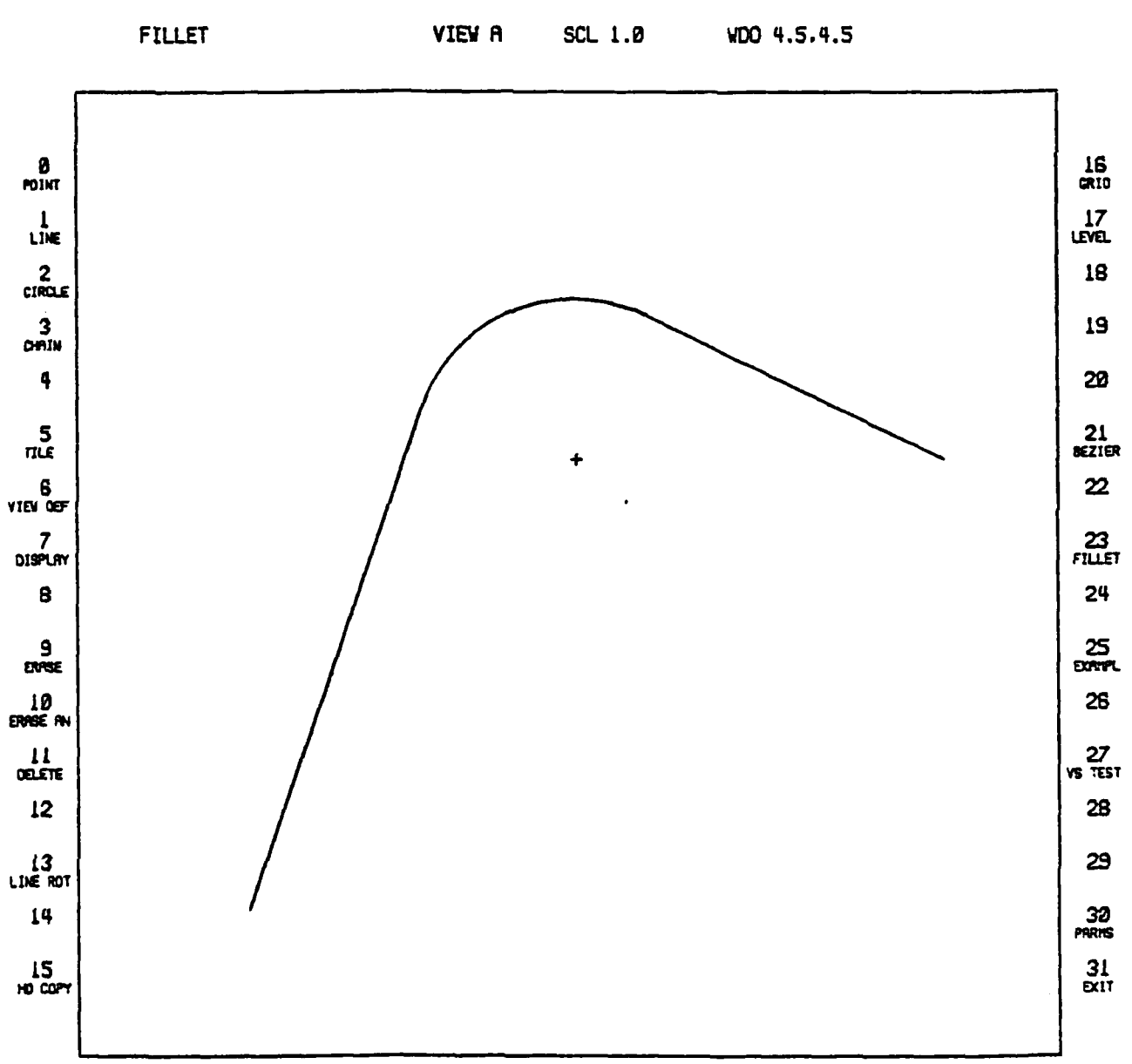
Key 23 - FILLET

This key will generate a circular fillet between two existing lines. The user selects two lines with the lightpen. The radius of the fillet is then entered via the keyboard in inches. The tracking cross is enabled so the user can indicate which of the four possible fillets should be generated. The fillet is then generated in a clockwise direction from the first line previously indicated to the second line. Construction lines are placed into a temporary file which is deleted when the function is terminated. A GP/1 point is generated at the center of the fillet. It is not deleted and may be used later.

At this point the initial lines entered are divided into two lines at the point of intersection with the fillet. The user can select any of the four lines to be deleted. This is typically used to trim off the excess part of the line extending beyond the fillet.

The user selects /DONE/ in the message field to terminate the function.


LINE
VIEW A
SCL 1.0
WDO 4.5.4.5
0 POINT
1 LINE
2 CIRCLE
3 CHAIN
4
5 TILE
6 VIEW DEF
7 DISPLAY
8
9 ERASE
10 ERASE AN
11 DELETE
12
13 LINE ROT
14
15 HD COPY
16 GRID
17 LEVEL
18
19
20
21 BEZIER
22
23 FILLET
24
25 EXAMPL
26
27 VS TEST
28
29
30 PARMS
31 EXIT


FILLET
VIEW A
SCL 1.0
WDO 4.5.4.5
0 POINT
1 LINE
2 CIRCLE
3 CHAIN
4
5 TILE
6 VIEW DEF
7 DISPLAY
8
9 ERASE
10 ERASE AN
11 DELETE
12
13 LINE ROT
14
15 HD COPY
16 GRID
17 LEVEL
18
19
20
21 BEZIER
22
23 FILLET
24
25 EXAMPL
26
27 VS TEST
28
29
30 PARMS
31 EXIT

APPENDIX B

GP/1 Work Session

Notes:
- <CR> is carriage return (the key labeled "NEW LINE")
- FK-0 is the function key labeled "f-0" in upper left hand corner of the keyboard.
- Lightpen sensitive "keys" along the edge of the screen will be indicated by "key no., label" or, for example "1-LINE". It should be picked with the lightpen.
- Lightpen sensitive items in the message field will be enclosed in "/" characters,"/POINTS/" for example.
- When entering a string of characters on the keyboard, the left arrow key (right side of the keyboard) will delete the last character and the "@" key will delete the entire line.

1. Login to your account

2. - Attach to User File Directory (UFD)
   SEG #GP/1_DEMO Start Program

   Figure 1 shows the screen format at the beginning of the program

3. Enter two points, first with the lightpen, second with the arrowkey :

   0 - POINT (Don't type this in! Select key 0 in the upper left area of screen with the lightpen.)

   /LIGHTPEN / (Again, select this with the lightpen in the message field at the top of the screen.)

   Move tracking cross with lightpen to position at or near point A in Figure 2.

   FK-0 - Confirm location
   0 - POINT
   /ARROWKEYS/ - Request entry using arrowkeys

   Use arrowkeys (right side of keyboard) to move cursor to location near point B in Figure 2

   FK-0 - confirm location

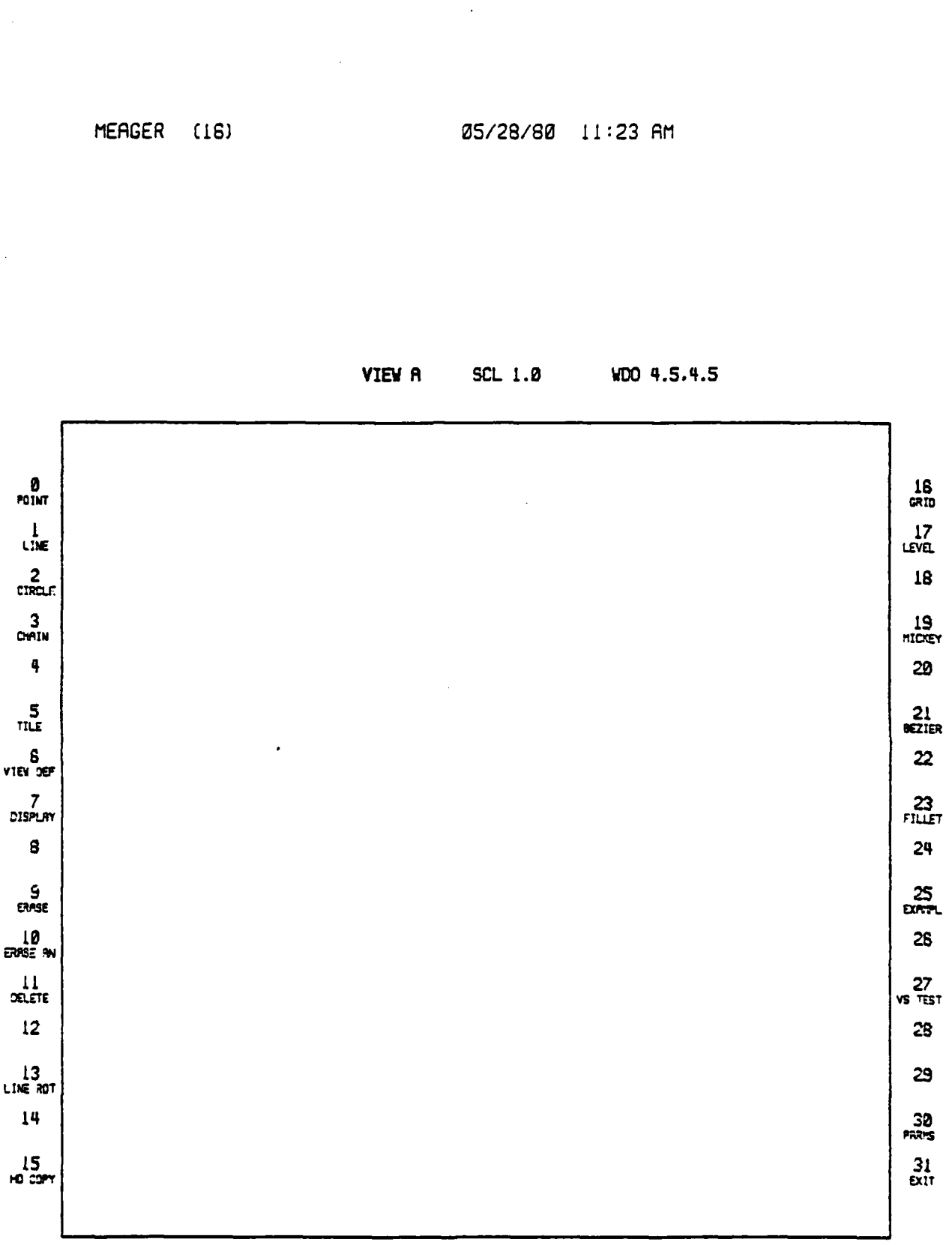


FIGURE 1

INITIAL SCREEN FORMAT FOR #PROG1

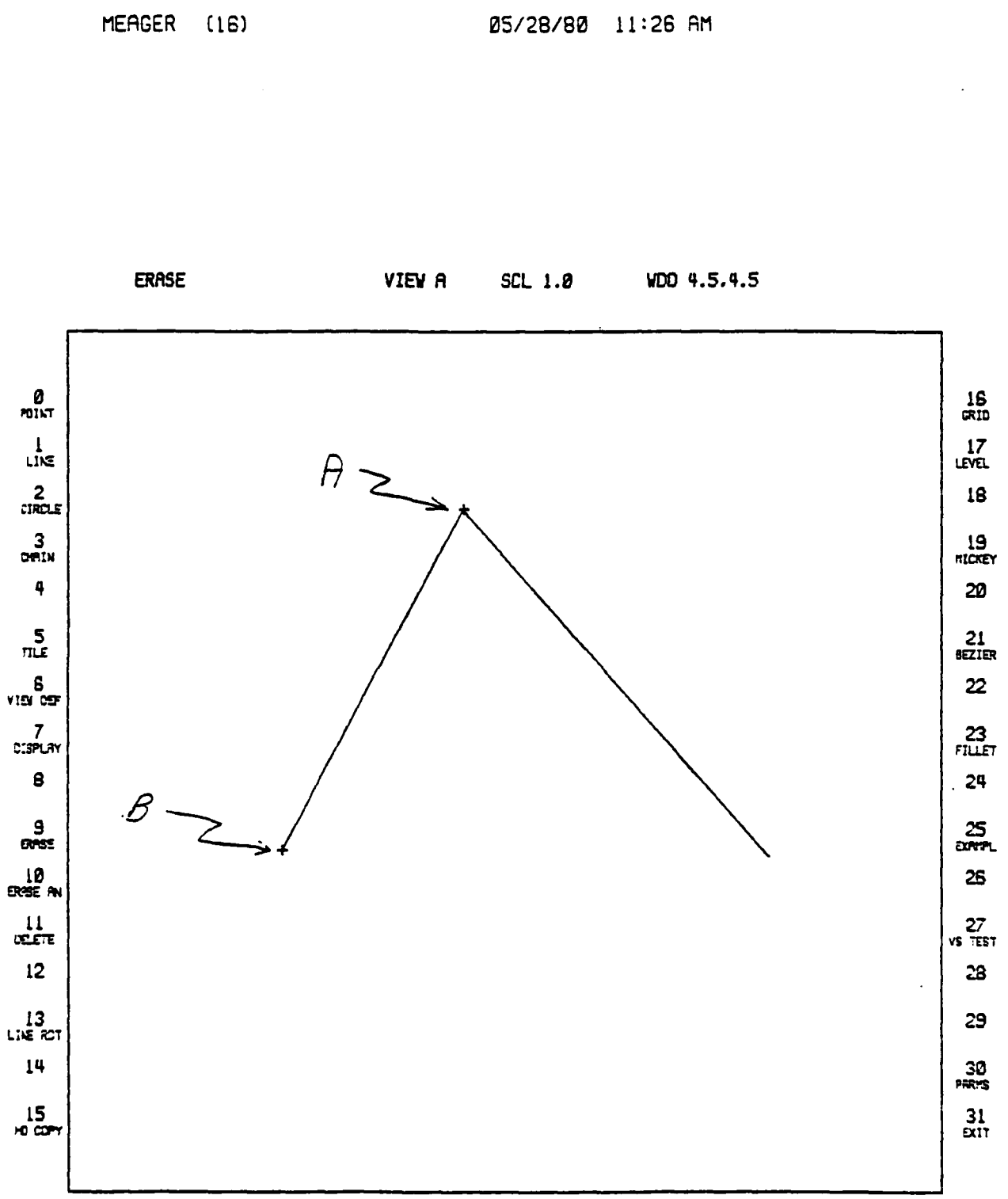


FIGURE 2

POINT AND LINE ENTRY EXAMPLE

4. Enter two lines, the first connecting the two points and the second from Point A to a second point entered with the lightpen:

1 - LINE Request line entry

/POINTS/ Request entry via two existing points

select one point and then the other with the lightpen (any order)

1 - LINE Request entry of second line

FIRST PT. /LP/ Request entry of first end point from lightpen.
(Note: there are two items labeled /LP/ in the message field. Select the one to the right.

move tracking cross with lightpen to lower end of right line in Figure 2.

FK - 0
/POINT/ Request entry of second end point from existing point

Select point A with lightpen

There should now be two lines on the screen as in Figure 2.

5. Draw a circular fillet from the left line to the right line in a clockwise direction with a radius of 1.5 inches. Also, trim off the excess parts of the lines. Enter as follows:

    23 - FILLET (with lightpen)

    select left line with lightpen

    select right line with lightpen

    1.5 <CR> - Enter requested radius of arc

    move tracking cross to any position below both lines to select desired fillet (center of screen is OK in this case)

    FK-0 - Confirm location

    The fillet should now be generated (see Figure 3). The two lower lines are construction lines. They will be removed.

    trim the excess section of each line (Hit part between fillet and Point A on each line with lightpen.)

    /DONE/ - Fillet function done

    The two shortened lines and the fillet should remain. Note also that the center of the fillet is left as a point which can be used later.

6. Generate two Bezier curves:

    first clear the items on the screen by repeating the following sequence for the 6 items on the screen:

        9 - ERASE

        select item with lightpen

    when all items have been cleared:

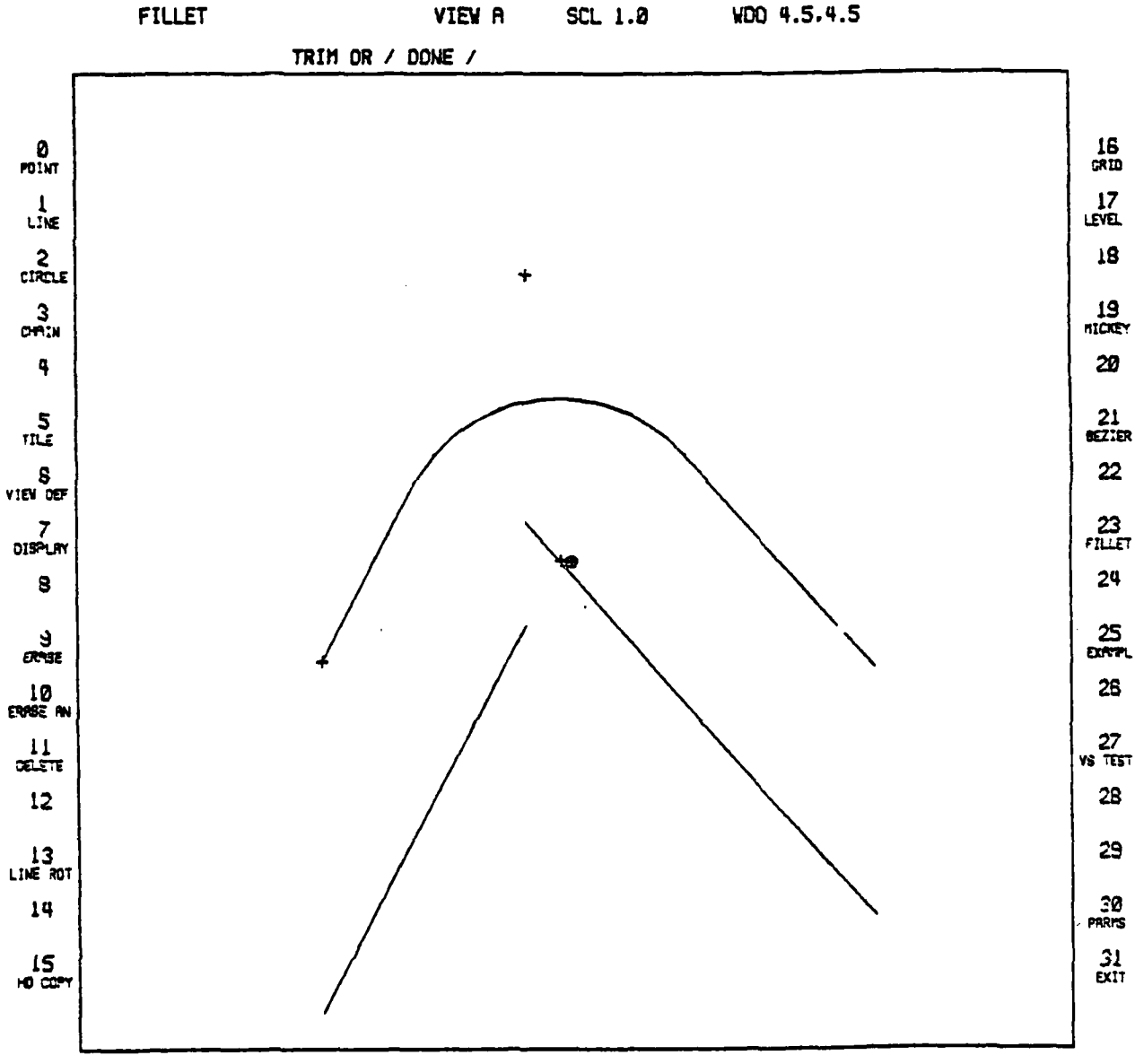


FIGURE 3

EXAMPLE OF FILLET FUNCTION

21 - BEZIER

move tracking cross to lower left hand point in Figure 4.

FK - 0

Wait for small marker to be put up at this location (under tracking cross). Move tracking cross and confirm about 15 or 20 points (maximum is 100) in a pattern similar to that in Figure 4. Confirm the last point twice.

A Bezier curve similar to Figure 4 should be generated after a short wait.

/MOVE POINT/ - Request movement of a control point

Select one of the points with the lightpen.

The tracking cross will appear at this location. Move it to the new location.

FK-0 - Confirm new location.

A new curve is generated.

/ADD POINT/

Select a point with the lightpen.

The tracking cross will appear at this point. Move it to the location of the new point.

FK-0

A new curve is generated with the new point in sequence between the selected point and its next neighbor.

/DEL POINT/ - Request deletion of a point.

Select point to be deleted with lightpen.

A new curve is generated without the selected point.

You may now move, add and delete points as desired. When satisifed with the curve, continue:

/ADD CURVE/ - Request a second curve.

Enter a small curve as above.

MEAGER (16) 05/28/80 11:29 AM

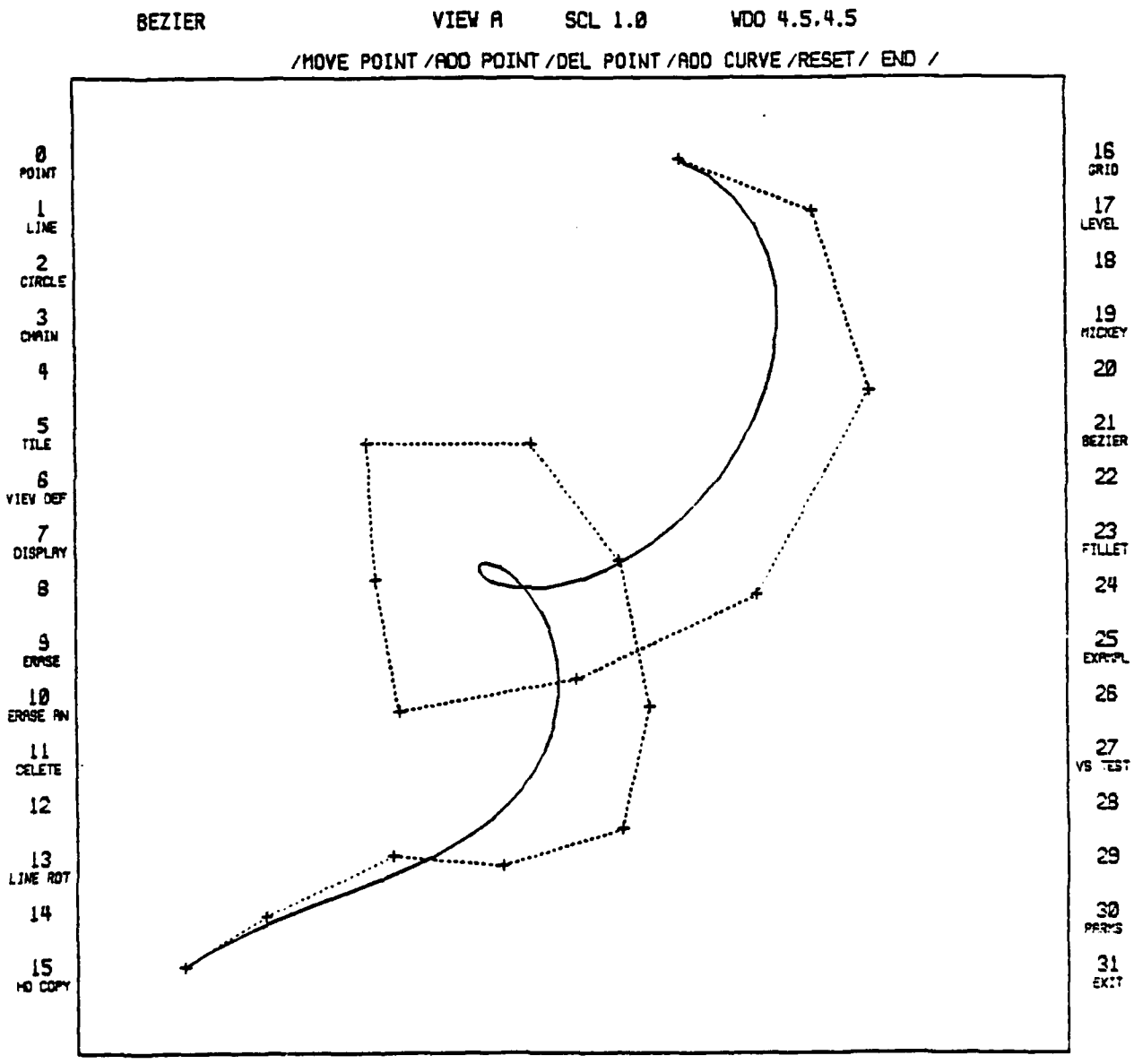


FIGURE 4

EXAMPLE OF BEZIER CURVE FUNCTION

/RESET/ - Clear the second curve and try again

The second curve will be deleted. Enter a new curve to generate a line similar to the left curve in Figure 5.

/END/ - Terminate Bezier curve function.

The control points and the construction lines will be removed, leaving the two curves. Generate a hardcopy as follows:

15 - HD COPY - Request hardcopy of screen

The image on the screen will be sent back to the PRIME computer for hardcopy generation. Note the "HARDCOPY ACTIVE" message in the upper right hand corner of the screen. Do not enter any requests until the message disappears.

Use 9 - ERASE to clear the screen of the two curves.

7. A drawing of a Mickey Mouse telephone is used to illustrate the concept of viewspace transformations:

19 - MICKEY - Request Mickey Mouse

/MAIN/ - Place in main viewspace

The image will be drawn line by line. See Figure 6.

9 - ERASE - Request erase

/MAIN/ - Request erase of main viewspace

6 - VIEW DEF - Request viewspace definition

/4/ - Select viewspace number 4

/VIEW CLIP/ - Select viewport clipping

The lightpen tracking cross will be enabled. Move it to the bottom left hand corner of a box about one-half the size of the screen. See Figure 7.

FK-0 - Confirm lower left hand corner

Move tracking cross to upper right hand corner of box.

FK-0

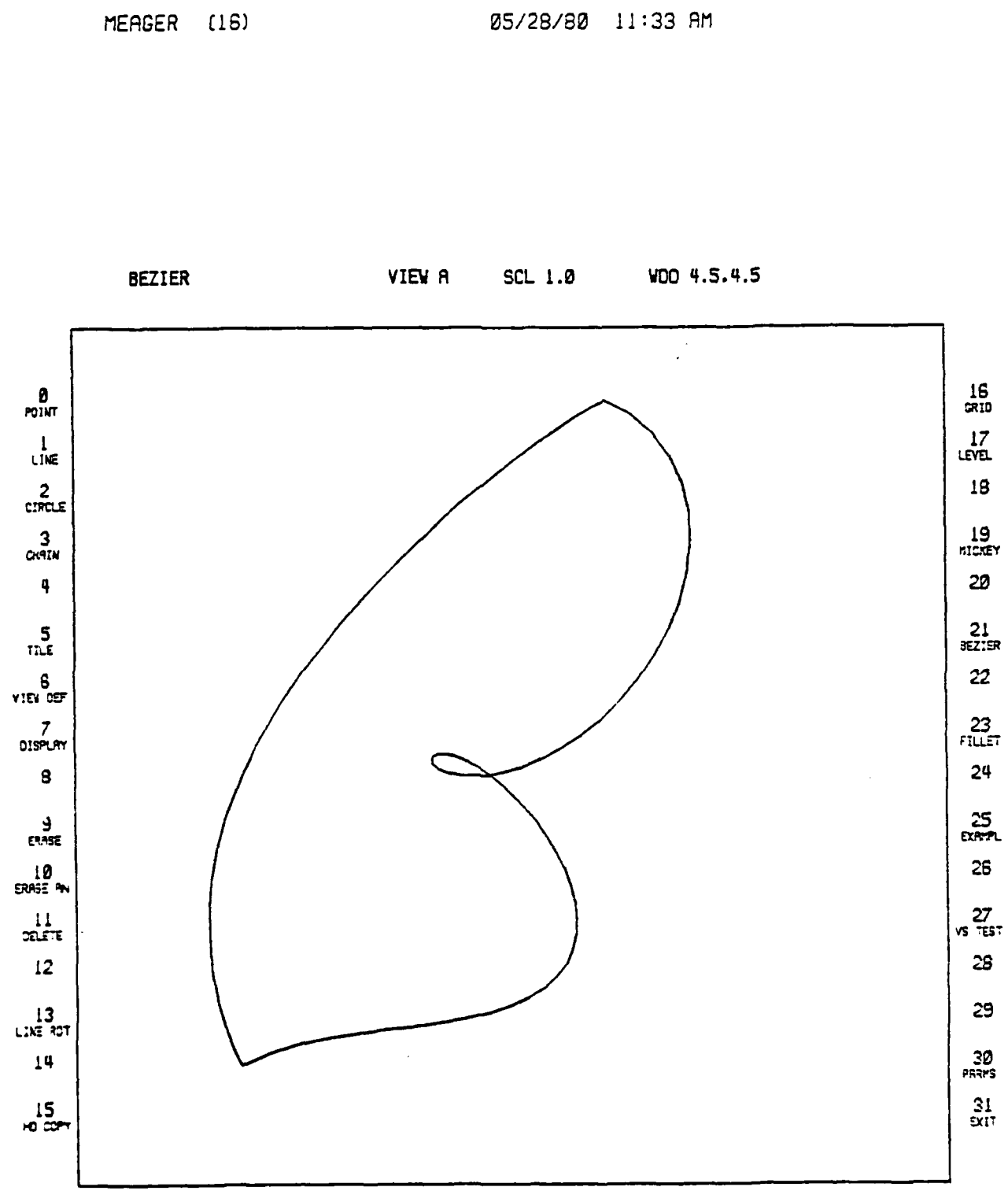


FIGURE 5

FINAL BEZIER DISPLAY

MEAGER (16) 05/28/80 11:35 AM

MICKEY VIEW A SCL 1.0 WDO 4.5.4.5

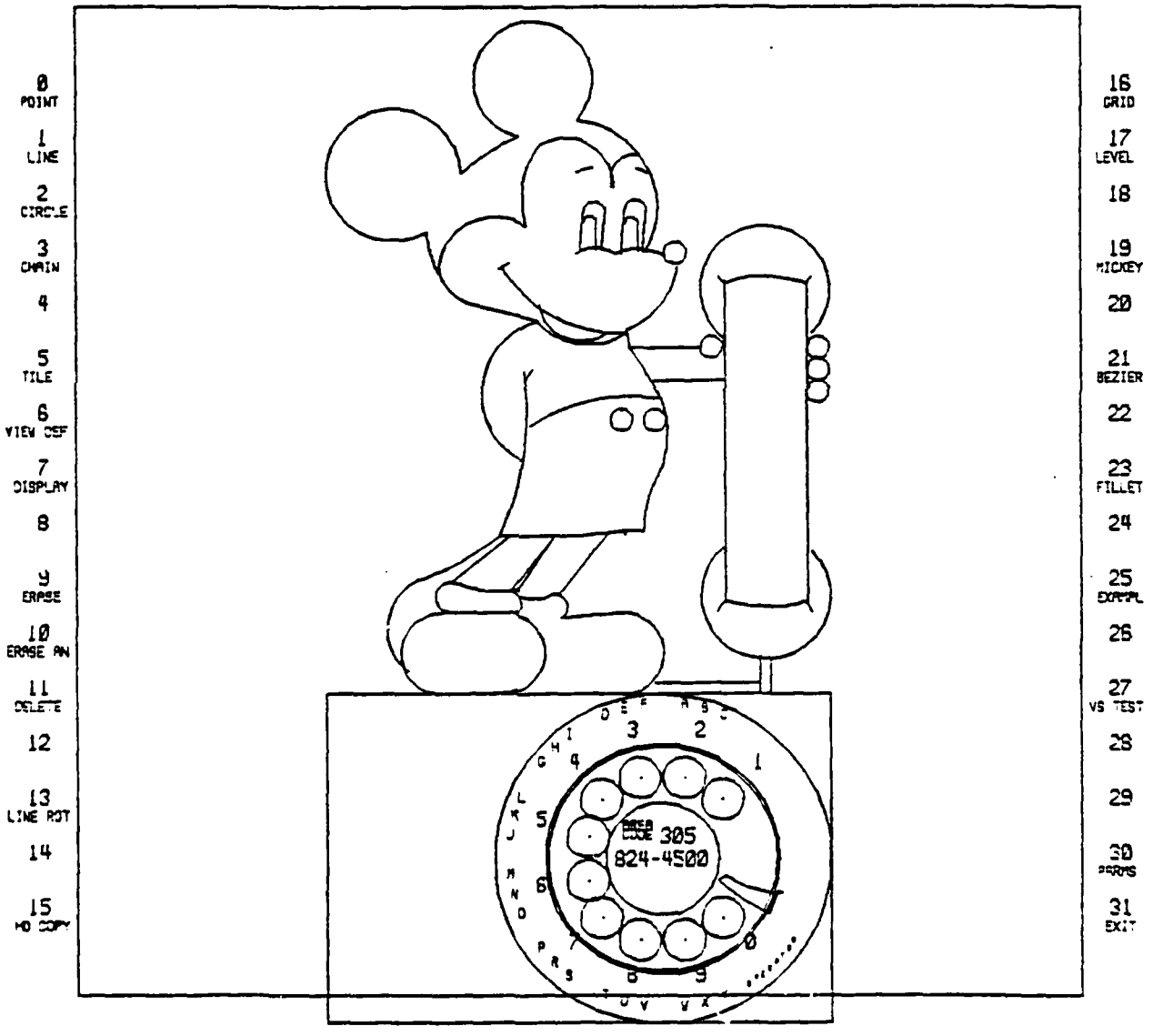


FIGURE 6

MICKEY IN VIEWPORT

MEAGER (16) 05/28/80 11:39 AM

MICKEY VIEW A SCL 1.0 WDO 4.5.4.5

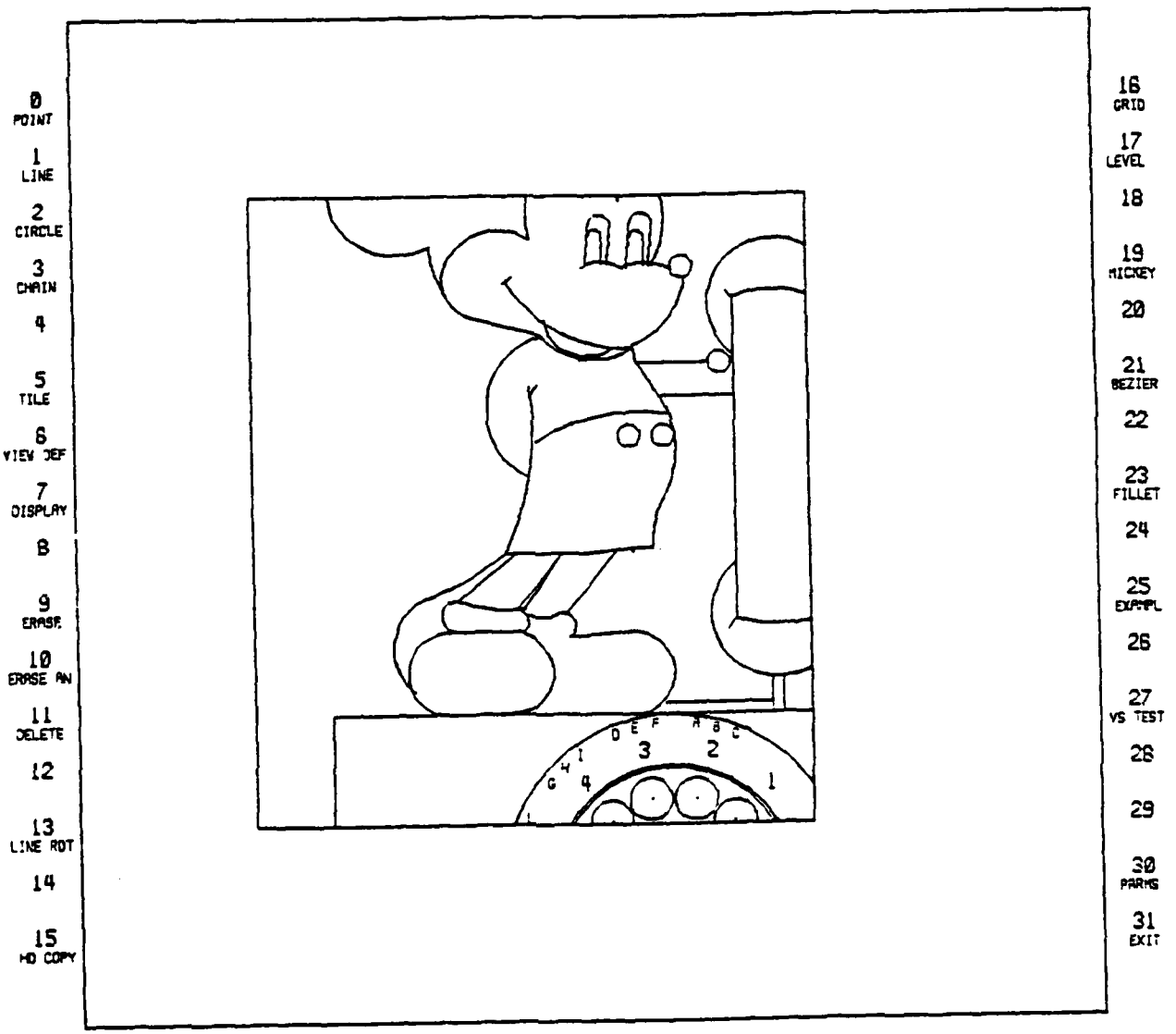


FIGURE 7

MICKEY IN MAIN VIEWSPACE

/STAT/ - Request change of status

VP CLIP: /ON/ - Request viewport clipping to be enabled

/CLOSE/ - Close viewspace definition

19 - MICKEY - Request Mickey

/4/ - Request Mickey to be displayed in viewspace number 4.

Note: The image will not be displayed line by line. It will be displayed completely after the transformations are completed. This may take a short time.

A view similar to Figure 7 should be displayed.

Now change the scale of Mickey:

6 - VIEW DEF - Request viewspace definition

/4/ - Request viewspace 4 again

/SCALE/ - Request change of scale

.4, .2 <CR> - Request scale down by 0.4 in X and 0.2 in Y

/STAT/ - Request change of status

GEOM: /ON/ - Enable geometric transformations

/CLOSE/ - Close viewspace

9 - ERASE - Erase

Hit Mickey with lightpen - Erase Mickey

19 - MICKEY

/4/ - Display Mickey with viewspace transformations

Mickey should now be displayed as in Figure 8. Note that alphanumeric characters are not scaled in size.

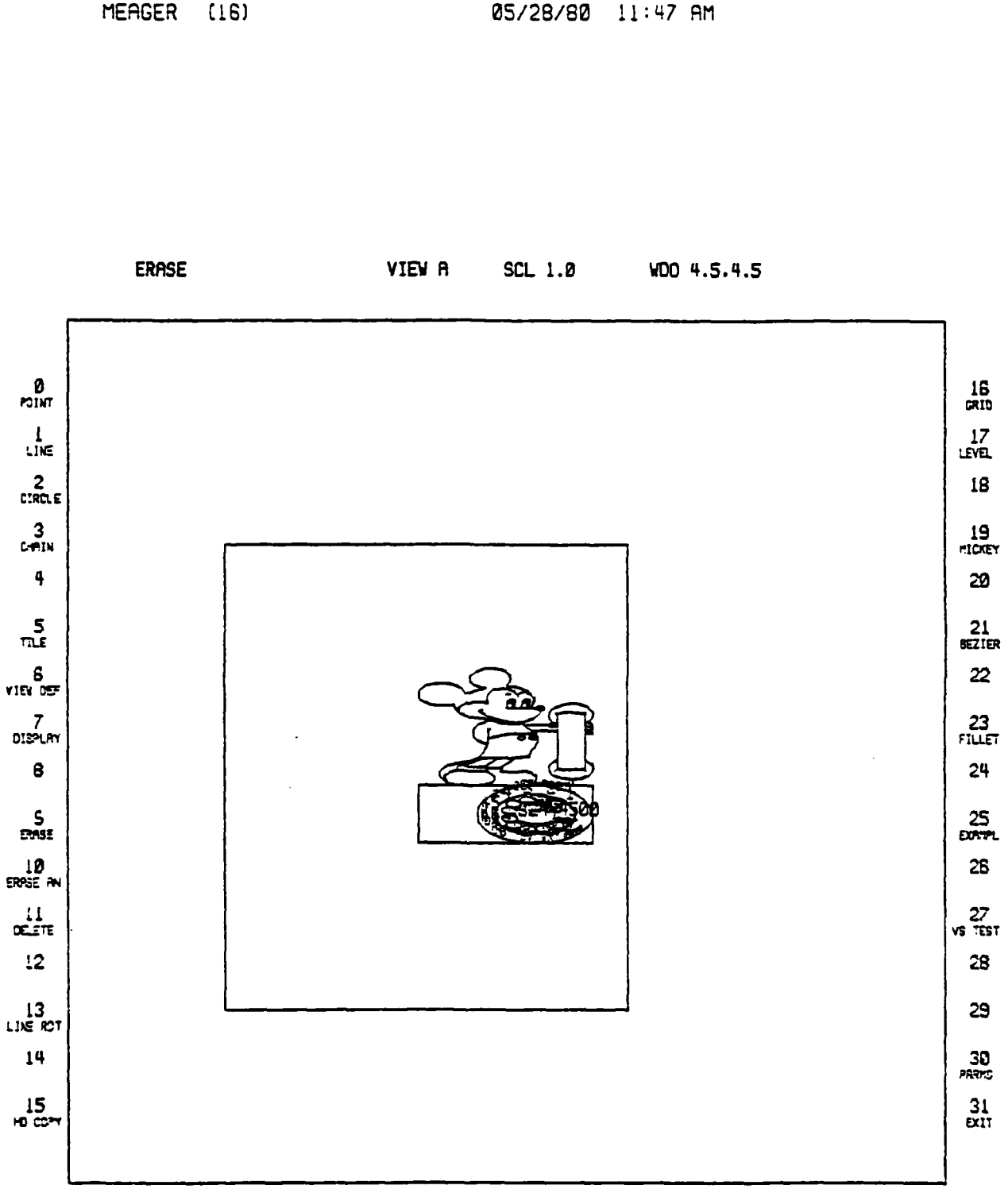


FIGURE 8

MICKEY SCALED 0.4 IN X AND 0.2 IN Y

Translate Mickey:

6 - VIEW DEF

/4/

/TRANS/ - Request translation

The lightpen tracking cross will be enabled at the center of the screen. Move it about 1 in. down and 1 in. to the left.

FK-O

/CLOSE/

19 - MICKEY

/4/

A second copy of Mickey should be displayed down and to the left of the original.

Erase the second copy of Mickey with 9 - ERASE.

Rotate Mickey:

6 - VIEW DEF

/4/

/ROT/ - Request rotation of viewspace

30<CR> - Request 30^{o} rotation

/CLOSE/

19 - MICKEY

/4/

A second copy of Mickey should be displayed down and to the left and rotated by 30^{o}.

Erase the second copy of Mickey with 9 - ERASE.

Window Clip Mickey:

6 - VIEW DEF

/4/

/WIND CLIP/ - Request clipping window

-3, 3, -2, 4<CR> - Request window clipping box from -3 to 3 in X and -2 to 4 in Y

/STAT/ - Request status change

WINDOW CLIP: /ON/ - Enable window clipping

/CLOSE/

19 - MICKEY

/4/

A rotated version of Mickey with most of the base of the original object eliminated and a small part of the top clipped is displayed. See Figure 9.

The viewport definition may now be modified until a desired image is generated. Additional viewspaces may be used.

8. Terminate the session

31 - EXIT Exit program

REALLY? /YES/ - Confirm exit

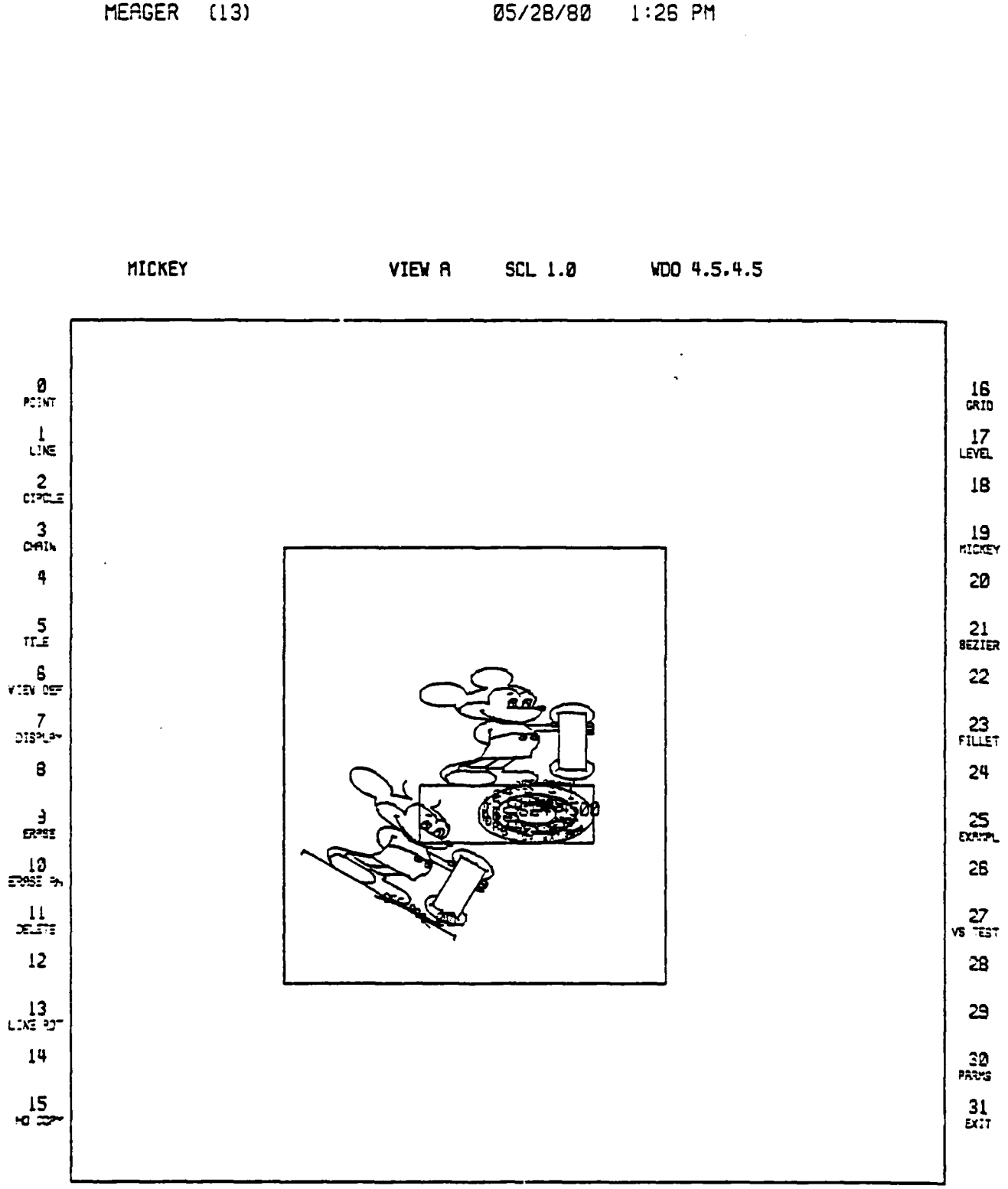


FIGURE 9

ROTATED MICKEY WITH WINDOW CLIPPING

Appendix C - GP/1 Main Program (GP/1_DEMO)

```
C
C  ##############################################################################

C
C
C     MAIN PROGRAM: GP/1_DEMO
C
C
C  PROGRAM NUMBER 0 - MAIN PROGRAM
C
C
C    COPYRIGHT 1979, MEAGER SOFTWARE LIMITED, LOUDONVILLE, NEW YORK
C
$INSERT GP/1.INSERT
C
C  GP/1 - FUNCTION KEY LABELS
C
      DATA FUN/
     0 '  POINT ',     /* LABEL FOR FUNCTION KEY NUMBER  0
     1 '   LINE ',     /* LABEL FOR FUNCTION KEY NUMBER  1
     2 '  CIRCLE',     /* LABEL FOR FUNCTION KEY NUMBER  2
     3 '  CHAIN ',     /* LABEL FOR FUNCTION KEY NUMBER  3
     4 '        ',     /* LABEL FOR FUNCTION KEY NUMBER  4
     5 '  TILE  ',     /* LABEL FOR FUNCTION KEY NUMBER  5
     6 'VIEW DEF',     /* LABEL FOR FUNCTION KEY NUMBER  6
     7 ' DISPLAY',     /* LABEL FOR FUNCTION KEY NUMBER  7
     8 '        ',     /* LABEL FOR FUNCTION KEY NUMBER  8
     9 '  ERASE ',     /* LABEL FOR FUNCTION KEY NUMBER  9
     0 'ERASE AN',     /* LABEL FOR FUNCTION KEY NUMBER 10
     1 ' DELETE ',     /* LABEL FOR FUNCTION KEY NUMBER 11
     2 '  IDENT ',     /* LABEL FOR FUNCTION KEY NUMBER 12
     3 '  SLICE ',     /* LABEL FOR FUNCTION KEY NUMBER 13
     4 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 14
     5 ' HD COPY',     /* LABEL FOR FUNCTION KEY NUMBER 15 (DON'T CHANGE!)
     6 '  GRID  ',     /* LABEL FOR FUNCTION KEY NUMBER 16
     7 ' LEVEL  ',     /* LABEL FOR FUNCTION KEY NUMBER 17
     8 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 18
     9 ' MICKEY ',     /* LABEL FOR FUNCTION KEY NUMBER 19
     0 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 20
     1 ' BEZIER ',     /* LABEL FOR FUNCTION KEY NUMBER 21
     2 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 22
     3 ' FILLET ',     /* LABEL FOR FUNCTION KEY NUMBER 23
     4 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 24
     5 ' EXAMPLE',     /* LABEL FOR FUNCTION KEY NUMBER 25
     6 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 26
     7 ' VS TEST',     /* LABEL FOR FUNCTION KEY NUMBER 27
     8 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 28
     9 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 29
     0 ' PARMS  ',     /* LABEL FOR FUNCTION KEY NUMBER 30
     1 '  EXIT  '/     /* LABEL FOR FUNCTION KEY NUMBER 31 (DON'T CHANGE!)
C
```

```
C  INITIALIZE TABLES
C
      CALL TINIT
C
C  INITIALIZE SCREEN
C
      CALL GRESET
      CALL ENTGRA
C
CCCCCCC     CALL SINIT
      CALL SVTITL(0)
C
C
C  GET FIRST LP HIT
C
99    CONTINUE
      CALL DSPFUN(-1)
C
       CALL SELECT(5,NSUB,ITYPE)
C
      KEY=ITYPE-FUNKEY
C
C  DECODE - CHECK TO SEE IF NOT FUNKEY FIRST
C
      IF((KEY.LT.0).OR.(KEY.GT.31)) GO TO 99
C
C  PUT KEY HIT IN THE UPPER LEFT HAND CORNER
C
      CALL DSPFUN(KEY)
C
C  CLEAR ACK, ERRORS, MESSAGES,  MENU (UNLESS HARDCOPY (KEY 15))
C
      IF(KEY.EQ.15) GO TO 8000
      CALL DSPERR(' ',0)
      CALL DSPACK(' ',0)
      CALL DSPMSG(' ',0)
      CALL DSPMNU(' ',0)
8000  CONTINUE
C
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,
     2 20,21,22,23,24,25,26,27,28,29,30,31), KEY
C
C  KEY 0 GOES HERE (DROPS THROUGH COMPUTED GO TO)
C  NOTE: USE ONLY STATEMENT NUMBERS 32 TO 98 IN HERE!!
C
      CALL FUN0             /* USER ENTRY OF A POINT
      GO TO 99
1     CONTINUE
      CALL FUN1             /* USER ENTRY OF A LINE
      GO TO 99
2     CONTINUE
      CALL FUN2               /* USER ENTRY OF A CIRCLE OR ARC
      GO TO 99
3     CONTINUE
```

```
      CALL FUN100
      GO TO 99
4     CONTINUE
      GO TO 99
5     CONTINUE
      CALL FUN343
      GO TO 99
6     CONTINUE
      CALL FUN6  /* CHANGE VIEWSPACE DEFINITION
      GO TO 99
7     CONTINUE
      GO TO 99
8     CONTINUE
      GO TO 99
9     CONTINUE
      CALL FUN3                 /* ERASE
      GO TO 99
10    CONTINUE
      CALL ERASA
      GO TO 99
11    CONTINUE
      CALL FUN4                    /* DELETE ITEM
      GO TO 99
12    CONTINUE
      CALL FUN201
      GO TO 99
13    CONTINUE
      CALL FUN411
      GO TO 99
14    CONTINUE
      GO TO 99
15    CONTINUE
C
C  *** NOTE: HARDCOPY IS IMPLEMENTED INSIDE SUBROUTINES SELECT β SELCT2.
C            THIS ALLOWS HARDCOPY TO BE GENERATED WHENEVER THE LIGHTPEN
C            IS ENABLED (EVEN IN THE MIDDLE OF A FUNCTION). DO NOT PUT
C            ANY CODE IN HERE (IT WOULD NEVER BE EXECUTED) AND DO NOT
C            CHANGE THE FUNCTION OF KEY 15!
C
      GO TO 99
C
16    CONTINUE
      CALL FUN102                  /* GRID
      GO TO 99
17    CONTINUE
      CALL FUN101
      GO TO 99
18    CONTINUE
      GO TO 99
19    CONTINUE
      GO TO 99
20    CONTINUE
      GO TO 99
21    CONTINUE
```

```
      CALL FUN341
      GO TO 99
22    CONTINUE
      GO TO 99
23    CONTINUE
      CALL FUN342
      GO TO 99
24    CONTINUE
      GO TO 99
25    CONTINUE
      CALL FUN340
      GO TO 99
26    CONTINUE
      GO TO 99
27    CONTINUE
      CALL FUN202
      GO TO 99
28    CONTINUE
      GO TO 99
29    CONTINUE
      GO TO 99
30    CONTINUE
      CALL FUN5                    /* CHANGE DEFAULT PARAMETERS
      GO TO 99
31    CONTINUE
C
C  NOTE: EXIT IS IMPLEMENTED IN SELECT AND SELCT2. THIS ALLOWS THE USER TO
C        EXIT WHENEVER THE LIGHTPEN IS ACTIVE. DO NOT PUT ANY CODE IN HERE
C        OR TRY TO CHANGE THE FUNCTION OF KEY 31.
C
      GO TO 99
C
      END
C
C
```

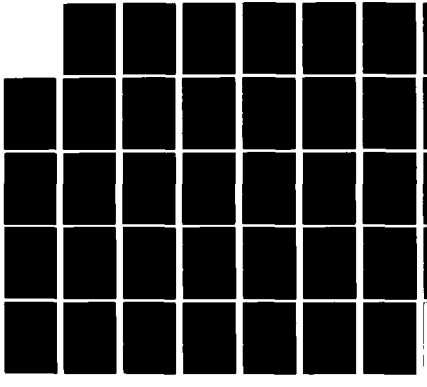
Appendix D - GP/1 Insert File (GP/1.INSERT)

```
C// GP/1.INSERT 203 GP/1 INSERT FILE
C
CI ------------------------------------------------------------------------
CI
CI     PROGRAM "GP/1" INSERT FILE
CI
CI IN THE SOURCE OF GP/1 PROGRAMS THIS FILE IS INSERTED WHEREVER THE
CI STATEMENT "$INSERT GP/1.INSERT" APPEARS BEGINNING IN COLUMN 1.
CI
CI ------------------------------------------------------------------------
CI
CI   COPYRIGHT 1979, MEAGER SOFTWARE LIMITED, LOUDONVILLE, NEW YORK
CI
CI  BEGIN INSERT FILE
CI
CI
CI
      INTEGER POINT,LINE,CIRARC,CHAIN,AREA,FUNKEY,ERROR,ACK,MESSAG
      INTEGER MENU,FUNCTN,LEVEL,NLINES,GRID,TEMPIC
      INTEGER*4 IQUED
      INTEGER*4 DFSUB$,PROJ$
      INTEGER*2 ICHVTB,IARVTB
      LOGICAL NUMFRE
CI
CI
      DOUBLE PRECISION FUN(32)
CI
CI  PARAMETER STATEMENTS - DEFINE THE SIZE OF TABLES
CI
      PARAMETER (M$PIC=128)     /* MAXIMUM NUMBER OF SUBPICTURES
      PARAMETER (M$PNT=30)      /* MAXIMUM NUMBER OF POINTS
      PARAMETER (M$LIN=20)       /* MAXIMUM NUMBER OF LINES
      PARAMETER (M$CIR=10)       /* MAXIMUM NUMBER OF CIRCLES
      PARAMETER (M$CH=10)       /* MAXIMUM NUMBER OF CHAINS
      PARAMETER (M$CHV=10000)    /* MAXIMUM NUMBER OF CHAIN VALUES (LINKS)
      PARAMETER (M$SCN=50000) /* MAXIMUM SIZE OF SCAN TABLE
      PARAMETER (M$QUE=50000) /* MAXIMUM SIZE OF QUEUE
      PARAMETER (M$QD=50000)  /* MAXIMUM SIZE OF DOUBLE PRECISION QUEUE
CI
CI  PARAMETER STATEMENTS - DEFINE COLUMNS OF OBJECT TABLE (IARTB)
CI
      PARAMETER (HEADR$=1)       /* HEADER
      PARAMETER (LEVEL$=2)       /* LEVEL
      DIMENSION ISUBTB(M$PIC,2)                  /* SUBPICTURE TABLE (TYPE,ENTRY)
      DIMENSION IPTABL(M$PNT),PTABLE(M$PNT,3)    /*POINT TABLES (HEADER,X,Y,Z)
      DIMENSION ILINTB(M$LIN),FLINTB(M$LIN,6)    /* LINE TABLES (HEADER,X1,Y1,
CI                                                  Z1,X2,Y2,Z2)
      DIMENSION ICIRTB(M$CIR),CIRTB(M$CIR,5)     /* CIRCLE TABLES (HEADER,X,Y,R,A,B)
      DIMENSION ICHTB(M$CH,6)                    /* CHAIN TABLE (HEADER,LEVEL,
CI                                                  ISTART,NOELEM,IGEGNX,IBEGNY)
      DIMENSION ICHVTB(M$CHV)                    /* CHAIN VALUE TABLE
```

```
      DIMENSION ISCAN(M$SCN,2)          /* SCAN TABLE
      DIMENSION IQUEUE(M$QUE,2)         /* QUEUE
      DIMENSION IQUED(M$QD,2)           /* DOUBLE PRECISION QUEUE (INTEGER*4)
CI
CI
      DIMENSION ICHAR$(2)               /* TEMPORARY STORAGE FOR EIGHT A/N
CI                                         CHARACTERS (FOR ENCODE, ETC.)
      DIMENSION DFSUB$(8)               /* NAME OF DATA FILE SUB-UFD
      DIMENSION DIMAG$(8)               /* NAME OF IMAGE FILE SUB-UFD
      DIMENSION DGRPH$(8)               /* NAME OF GRAPHICS FILE SUB-UFD
      DIMENSION PROJ$(8)                /* NAME OF PROJECT (USED TO GENERATE
CI                                         NAMES OF DATA FILES)
      DIMENSION IBUF$(25)               /* TEMPORARY STORAGE FOR UP TO 100
CI                                         CHARACTERS (FOR READ, WRITE, ETC.)
CI
      COMMON/AREA/ISUBTB,IPTABL,PTABLE,ILINTB,FLINTB,ICIRTB,CIRTB,
     2 MAXPIC,POINT,LINE,CIRARC,IDMARK,FUNCTN,ERROR,ACK,MESSAG,MENU,
     3 FUNKEY,FUN,CHAIN,AREA,LEVEL,USCALE,XORIGN,YORIGN,NLINES,
     4 ICHTB,IARTB,MAXCH,MAXCHV,MAXAR,MAXARV,
     5 MAXSCN,LEVSCN,NPOINT,NHASH,HDELTA,IQR,IQF,MAXQUE,IGRID,
     6 GRID,NDELAY,INTENS,LCHINS,TEMPIC,IQFD,IQRD,MAXQD,PI,ITABLT,
     7 COL$PW,COL$OF,DFSUB$,PROJ$,I$PART,IBACK,MAXCIR,MAXLIN,MAXPNT,
     8 GSMLT$,SLMLT$,IBCK$0,IBCK$1,IBCK$2,IBUF$,DIMAG$,DGRPH$
CI
      COMMON/CHV/ICHVTB
CI
      COMMON/QUE/IQUEUE
      COMMON/QUE2/IQUED
      COMMON/SCANCM/ISCAN
CI
CI
      DATA MAXPIC/M$PIC/     /* MAXIMUM NUMBER OF SUBPICTURES
      DATA MAXPNT/M$PNT/     /* MAXIMUM NUMBER OF POINTS
      DATA MAXLIN/M$LIN/     /* MAXIMUM NUMBER OF LINES
      DATA MAXCIR/M$CIR/     /* MAXIMUM NUMBER OF CIRCLES
      DATA MAXCH/M$CH/       /* MAXIMUM NUMBER OF CHAINS
      DATA MAXCHV/M$CHV/     /* MAXIMUM NUMBER OF CHAIN VALUES (LINKS)
      DATA MAXSCN/M$SCN/     /* MAXIMUM SIZE OF SCAN TABLE
      DATA MAXQUE/M$QUE/     /* MAXIMUM SIZE OF QUEUE
      DATA MAXQD /M$QD/      /* MAXIMUM SIZE OF DOUBLE PRECISION QUEUE
CI
CI  DEFINITION OF SUBPICTURE TYPES
CI
      DATA POINT/1/          /* POINT
      DATA LINE/2/           /* LINE
      DATA CIRARC/3/         /* CIRCLE/ARC
      DATA CHAIN/4/          /* CHAIN ENCODED BLOB
      DATA FUNKEY/100/       /* FUNCTION KEY 0
      DATA IBACK/150/        /* BACKGROUND SUBPICTURE
      DATA ERROR/155/        /* ERROR MESSAGE
      DATA ACK/156/          /* ACKNOWLEDGEMENT MESSAGE
      DATA MESSAG/170/       /* MESSAGE (FIRST)
      DATA MENU/185/         /* MENU (FIRST)
      DATA FUNCTN/151/       /* FUNCTION MESSAGE
```

```
      DATA GRID/161/       /* GRID LINES
      DATA TEMPIC/250/     /* TEMPORARY SUBPICTURE (CAN BE DELETED OR ERASED)
CI
CI  DEFAULTS
      DATA IDMARK/4/           /* DEFAULT POINT MARKER (4=CROSS)
      DATA LEVEL/5/            /* DEFAULT LEVEL
      DATA NLINES/32/          /* DEFAULT 2**LEVEL VALUE
      DATA XORIGN,YORIGN/-4.5,-4.5/  /* DEFAULT ORIGIN
      DATA USCALE/1.75781/     /* DEFAULT UNIVERSAL SCALE (9 IN VIEWABLE/(1024/200))
      DATA LEVSCN/1/           /* LEVEL REPRESENTED BY THE VALUES CURRENTLY IN
CI                                THE SCAN TABLE ISCAN
      DATA NPOINT/0/           /* NUMBER OF POINTS IN THE SCAN TABLE
      DATA NHASH/3/            /* LEVEL OF GRID LINES (POINTS BETWEEN = 2**NHASH)
      DATA IQF/0/              /* FRONT QUEUE POINTER
      DATA IQR/0/              /* REAR QUEUE POINTER
      DATA IQFD/0/             /* DOUBLE PRECISION QUEUE - FRONT POINTER
      DATA IQRD/0/             /* DOUBLE PRECISION QUEUE - REAR POINTER
      DATA LCHINS/10/          /* DEFAULT INTENSITY FOR CHAIN DISPLAY
      DATA PI/3.14159/         /* THE VALUE OF PI
      DATA DFSUB$/'S_OCTREE_DATA_FILES              '/ /* NAME OF DATA SUB-UFD
      DATA DIMAG$/'S_OCTREE_IMAGE_FILES             '/ /* NAME OF IMAGE SUB-UFD
      DATA DGRPH$/'S_OCTREE_GRAPHICS_FILES          '/ /* NAME OF GRAPHICS SUB-UFD
      DATA PROJ$ /'D_MISC                           '/ /* PROJECT NAME
CI
C********************************************************* END OF BLOCK *******
C
```

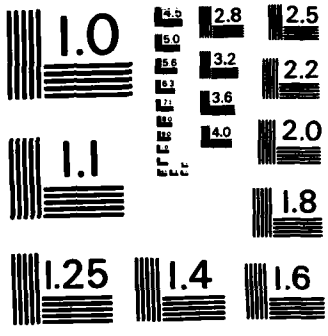
AD-A121 405 DOCUMENTATION MANUAL GRAPHICS PACKAGE NUMBER 1 (GP/1) 2/2
AND PROGRAM OCTREE(U) RENSSELAER POLYTECHNIC INST TROY
NY IMAGE PROCESSING LAB D MEAGHER AUG 82 IPL-TR-028
UNCLASSIFIED N00014-82-K-0301 F/G 9/2 NL



END

FILMED

DTIC



APPENDIX E

Description of Most Commonly Used GP/1 Subroutine

SOME COMMON GP/1 ABBREVIATIONS

DSP - Display

ACK - Acknowledgement

ERR - Error

MSG - Message

MNU - Menu

LP - Lightpen

IN - Input

FUN - Function

OT - Output

PIC - Subpicture

PNT - Point

LIN - Line

CIR - Circle/Arc

TB - Table

PUT - Put (SOMETHING INTO A TABLE)

GET - Get (SOMETHING FROM A TABLE)

DEL - Delete (SOMETHING FROM A TABLE)

TEMP - Temporary

SUB - Subpicture

AK - Arrow Keys

KEY - Function Keys

DISPLAY TEXT IN FIELDS

Display Acknowledgements

CALL DSPACK ('TEXT)', NCHAR)

(TEXT) - text of the acknowledgement message

NCHAR - number of characters up to 23

NOTES:

(1) Any existing acknowledgement is deleted

(2) NCHAR = 0 will clear any existing acknowledgement

```
EX:  Call DSPACK ('LINE DELETED' , 12)
     Call DSPACK(' ',0)     /* CLEAR ACK FIELD
```

Display Error Message

CALL DSPERR ('(TEXT)', NCHAR)

(TEXT) - text of error message

NCHAR - number of characters (up to 57)

NOTES:

(1) The message will blink

(2) Any existing error message is cleared

(3) NCHAR = 0 will clear any existing error message

(4) A LP hit on the error field will clear the error message

(5) The error message is right justified

```
EX:  Call DSPERR ('QUEUE UNDERFLOW. OPERATION ABORTED, 35)
```

Display Message(s)

CALL DSPMSG ('(TEXT)', NCHAR)

(TEXT) - Text of message

NCHAR - Number of characters (up to 71 for all messages)

NOTES:

(1) Existing messages are NOT cleared. New message is placed to the right of exsiting messages.

(2) Clear ALL existing messages if NCHAR=0

(3) Up to 15 messages can be displayed at one time and each is independently LP selectable.

EX: Call DSPMSG ('PLEASE INDICATE THE FIRST LINE WITH THE LIGHTPEN', 48)

Display Menu Items

CALL DSPMNU ('(TEXT)', NCHAR)

(TEXT) - Text of the message

NCHAR - Number of characters (up to 66 for all menu items)

NOTES:

(Same as DSPMSG)

EX: Call DSPMSG ('/POINT ENTRY FROM KEYBOARD/', 27)

SELECTING AN ITEM ON THE SCREEN

GP/1 is designed to encourage the use of the light pen for most operator interaction with the applications program. Thus, most screen items are LP sensitive (points, lines, circles, menu items, messages, function keys, etc.).

The following subroutine is called for item selection:

CALL SELECT (ICALL, NSUB, ITYPE)

ICALL - subroutine call number (any integer *2, numbers, see below)

NSUB - number of subpicture (1 to 128)

ITYPE - the item type (see below)

ITYPE

The integer variable ITYPE indicates the type of item selected. The system insert file defines the value of a number of variables to test ITYPE. They are currently:

POINT - if ITYPE .EQ. POINT the selected subpicture is a point

LINE - if ITYPE .EQ.LINE the subpicture is a line

CIRCLE - if ITYPE .EQ. CIRCLE the subpicture is a circle or arc

CHAIN - if ITYPE .EQ. CHAIN the subpicture is a chain encoded blob.

MESSAG - if ITYPE .EQ. MESSAGE + N the subpicture is message N (starting from 0)

MENU - if ITYPE .EQ. MENU + N the subpicture is menu item N (starting from 0)

FUNKEY - if ITYPE .EQ. FUNKEY + N the subpicture is Key N (starting from 0) .

TEMPIC - Temporary subpicture type defined for general purpose use.

EMPTY - A value of 0 used to indicate empty table entries (see below). Not normally used by applications programmers.

GP/1 POINT INPUT SUBROUTINES

Lightpen

The following will enable the tracking cross at the center of the screen for lightpen input of one point (confirm with FK-0):

```
CALL LPIN (X,Y)

     (X,Y) - location entered by user
             (default screen coordinates)
```

To initialize the tracking cross at a specified location on the screen:

```
CALL LPIN2 (X,Y)


     (X,Y) - location entered by user
             (default screen coordinates)
```

Arrow Keys

The following will enable the crosshair at the center of the screen for arrow key input of one point (confirm with FK-0):

```
CALL AKIN (X,Y)

     (X,Y) - location entered by user
             (default screen coordinates)
```

To initialize the crosshair at a specified location:

```
CALL AKIN2 (X,Y)


     (X,Y) - location entered by user
             (default screen coordinates)
```

SUBPICTURE SUBROUTINES

Obtain a Subpicture

Obtain an available subpicture from SUBPICTURE TABLE and reserve.

CALL GETSUB (ICALL,NSUB,ITYPE)

ICALL - call number
NSUB - subroutine number (returned)
ITYPE - type of item subpicture reserved for (must be specified in call)

Notes:

(1) Normally, users will specify the ITYPE to be TEMPIC for a temporary subpicture for general use. The subpicture so generated may be deleted by the currently defined DELETE & ERASE keys. Any call to GETSUB with another ITYPE must be done carefully to avoid destroying the integrity of the GP/1 system tables (generating subroutine entries for nonexistent lines, for example).

(2) All subpictures are displayed at (o,o) at system initialization. Do not call DSPICT!

(3) A request for a type EMPTY or 0 subpicture generates a fatal error.

EX: CALL GETSUB (29100,NSUB,TEMPIC)

Subpicture NSUB has now been reserved as a temporary subpicture for general use. It can be deleted by the user with keys DELETE and ERASE or with subroutine DELSUB (below).

Delete a Subpicture

To delete a subpicture:

CALL DELSUB (ICALL,NSUB)

ICALL - call number
NSUB - subroutine number to be deleted

Notes:

(1) If the subpicture is undefined in the SUBPICTURE TABLE a non-fatal warning is generated.

(2) The subpicture is cleared and returned to empty status in the SUBPICTURE TABLE.

(3) If the subpicture represents a known item (point, line, etc.) the corresponding item table entries are cleared.

Search for a Subpicture Type

The following call will search the SUBPICTURE TABLE for a subpicture of type "ITYPE" beginning with subpicture "ISTART".

CALL SERCH (NSUB,ITYPE,ISTART)

NSUB - subpicture number (returned)
ITYPE - subpicture type to search for
ISTART- the number of the first subpicture to be examined (1 to MAXPIC)

Notes:

(1) If no subpicture of type ITYPE is found, NSUB is returned with a value of 0.

(2) The value of ISTART is not modified by SEARCH. The calling subroutine must step the value after a find if the next is desired (set ISTART = NSUB+1).

(3) The value of ISTART should not exceed the value of MAXPIC.

POINT, LINE & CIRCLE/ARC SUBROUTINES

Point

- To insert a point into the GP/1 tables (SUBPICTURE TABLE & POINT TABLE) and display in a new subpicture:

```
CALL PUTPNT (ICALL,NSUB,IHEADR,XVAL,YVAL)

     ICALL - call number
     NSUB  - subpicture number of new subpicture point put into
             (returned by subroutine)
     IHEADR- user defined header to identify point (can be any *2
             integer except 0)
     (XVAL,YVAL) - location of point
```

Notes:

(1) The POINT TABLE entry number can be found in location ISUBTB (NSUB,2).

(2) The new point is in an independent subpicture and can be selected with a SELECT call, erased with the ERASE key, deleted from the tables with the DELETE key, or DELSUB subroutine, etc.

- To read a point from the GP/1 tables:

```
CALL GETPNT (ICALL,NSUB,IHEADR,XVAL,YVAL)

     ICALL - call number
     NSUB  - subpicture number (specified)
     IHEADR- point header (returned)
     (XVAL,YVAL) - location of point (returned)
```

Notes:

(1) Errors are generated if NSUB is not defined or NSUB does not represent a point.

(2) This subroutine simply reads the parameters of the point. Nothing is changed.

Lines

- To insert a line into the GP/1 tables (SUBPICTURE TABLE & LINE TABLE) and display in a new subpicture:

```
CALL PUTLIN (ICALL,NSUB,IHEADR,X1,Y1,X2,Y2,TVAL)

     ICALL - call number
     NSUB  - subpicture number (returned)
     IHEADR- line header value (not 0) specified by user.
     (X1,Y1) location of first end point
     (X2,Y2) location of second end point
     TVAL  - (reserved for future use)
```

Notes:
(1) The LINE TABLE entry can be found in location ISUBTB (NSUB,2)
(2) See note (2) under PUTPNT.

- To obtain a line from the GP/1 tables:

```
CALL GETLIN (ICALL,NSUB,IHEADR,X1,Y1,X2,Y2,TVAL)

     ICALL - call number
     NSUB  - subpicture number (specified by calling routine)
     IHEADR- line header value (returned)
     (X1,Y1) location of first endpoint (returned)
     (X2,Y2) location of second endpoint (returned)
     TVAL  - (reserved for future use)
```

Notes:
(1) An error is generated if NSUB is not defined or is not a line.
(2) Nothing is modified by this call.

Circle/Arc

- To insert a circle or arc into the GP/1 tables and display in a new subpicture:

```
CALL PUTCIR (ICALL,NSUB,IHEADR,XVAL,YVAL,RADIUS,BEGANG,ENDANG)

     ICALL  - call number
     NSUB   - subpicture number (returned)
     IHEADR - circle/arc header (specified)
     (XVAL,YVAL) - center of circle or arc
     RADIUS - radius of circle or arc
     BEGANG - start angle relative to positive X axis in radians
     ENDANG - end angle relative to positive X axis in radians
              (arc drawn counterclockwise)
```

Notes:
(1) The CIRCLE/ARC TABLE entry can be found in location ISUBTB (NSUB,2).
(2) See note (2) under PUTPNT.

- To obtain a circle or arc from the GP/1 tables:

```
CALL GETCIR (ICALL,NSUB,IHEADR,XVAL,YVAL,RADIUS,BEGANG,ENDANG)

     ICALL  - call number
     NSUB   - subpicture number (specified)
     IHEADR - circle/arc header (returned)
     (XVAL,YVAL) - center (returned)
     RADIUS - radius (returned)
     BEGANG,ENDANG - beginning and ending angles (returned)
```

Notes:
(1) An error is generated if NSUB is not defined or is not a circle or arc.
(2) Nothing is modified by this call.

CHAIN SUBROUTINES

- To input a chain via lightpen or tablet and display it (as it is being generated) in a new subpicture:

```
CALL CHIN (ICALL,NSUB,IHEADR)

     ICALL - call number
     NSUB  - subpicture number into which the chain is placed
             (returned)
     IHEADR- header number of chain (not 0) (specified)
```

Notes:

(1) If the user confirms the root of the system above the GP/1 workspace, chain input will be transferred to the graphics tablet (SYSA only!)

(2) The new chain has an ITYPE of CHAIN and can be erased, deleted, etc.

(3) The chain number can be found in ISUBTB (NSUB,2)

- To display a chain in a new subpicture:

```
CALL DSPCH (NCHAIN)

     NCHAIN - chain number (entry in CHAIN TABLE) (specified)
```

SUPPORT SUBROUTINES FOR CHAIN CODES

Two chain code processing subroutines have been included in GP/1.SUBS. They are:

SCAN - This subroutine will work through the chain and fill a scan table with the X,Y locations of the links.

SORT - This subroutine will sort the chain scan table by X and Y value.

The sorted scan table is useful in determining the area of the shape represented by the chain, determining if a point is inside the boundary, and so on.

Appendix F - GP/1 Subroutines

SOME OF THE VARIABLES DEFINED FOR UNIVERSAL USE ARE:

ICALL - THE CALL NUMBER OF THE SUBROUTINE CALL. IN SOME OF THE LOWER LEVEL SUBROUTINES, THIS IS USED TO IDENTIFY THE SPECIFIC CALL TO A SUBROUTINE IF AN ERROR OCCURS (IT'S PRINTED OUT).THE CONVENTION IS TO USE AN INTEGER.
THE FIRST FOUR DIGITS REPRESENTING THE SUBROUTINE NUMBER AND THE LAST TWO REPRESENTING A SPECIFIC CALL WITHIN THE SUBROUTINE.
WHEN FIRST WRITING A SUBROUTINE, SEQUENCE BY 10 (ALWAYS MAKE THE LAST DIGIT A ZERO). THIS ALLOWS SPACE TO ADD CALL NUMBERS LATER.

NSUB - THE SUBROUTINE NUMBER. THIS IS USED TO IDENTIFY A SPECIFIC ITEM SUCH AS A POINT, LINE, MENU ITEM, ETC. CURRENTLY NSUB CAN RANGE BETWEEN 1 AND 128. IF THE PROGRAM GROWS BEYOND THESE LIMITS, NSUB COULD INCREASE BY PUTTING ALL ITEMS SUCH AS POINTS INTO ONE SUBPICTURE AND DETERMING NSUB BY LOCATION.

ITYPE - THE ITEM TYPE REPRESENTED BY A SUBPICTURE. IN GENERAL THE ACTUAL VALUE IS NOT TESTED FOR BUT RATHER A VARIABLE DEFINED IN THE INSERT FILE IS USED. IN SOME CASES A BASE NUMBER IS DEFINED. SUBSEQUENT ITEM ARE INCREMENTS FROM THE BASE. FOR EXAMPLE, UP TO 15 ITEMS CAN BE PUT UP AS MENU ITEMS. THE FIRST ITEM PUT UP (AFTER THE MENU HAS BEEN CLEARED) HAS AN ITYPE EQUAL TO THE VALUE OF THE INTEGER VARIABLE CALLED 'MENU'. WHEN A SUBPICTURE HAS BEEN SELECTED WITH THE LIGHT PEN, ITS ITYPE COULD BE COMPARED TO 'MENU' TO DETERMMINE IF IT WAS THE FIRST MENU ITEM WHICH WAS SELECTED. THE SECOND MENU ITEM DISPLAYED (SECOND CALL TO THE DISPLAY MENU SUB-ROUTINE) HAS AN ITYPE OF MENU+1. THE THIRD IS MENU+2, AND SO ON. THE ITYPE OF THE SELECTED SUBPICTURE IS THUS TESTED TO DETERMINE THE ACTION DESIRED BY THE USER.
THE CONVENTION IS TO ENCLOSE SEPARATE MENU AND MESSAGE ITEMS IN SLASH ('/') CHARACTERS ('/ITEM 1/ITEM 2/...').
THE FOLLOWING ARE CURRENTLY DEFINED:

- POINT - IF ITYPE.EQ.POINT, THE SUBPICTURE IS A POINT
- LINE - IF ITYPE.EQ.LINE, THE SUBPICTURE IS A LINE
- CIRCLE - IF ITYPE.EQ.CIRCLE, THE SUBPICTURE IS A CIRCLE OR ARC
- CHAIN - IF ITYPE.EQ.CHAIN, THE SUBPICTURE IS A BLOB REPRESENTED BY A FREEMAN CHAIN CODE
- AREA - IF ITYPE.EQ.AREA, THE SUBPICTURE IS A BLOB REPRESENTED BY AN AREA CODE
- FUNKEY - IF ITYPE.EQ.FUNKEY+N, THE SUBPICTURE REPRESENTS FUNCTION KEY N (0 TO 31)
- ERROR - ERROR MESSAGE
- ACK - ACKNOWLEDGEMENT MESSAGE
- MESSAG - IF ITYPE.EQ.MESSAG+N, THE SUBPICTURE REPRESENTS MESSAGE NUMBER N (STARTING FROM 0).
- MENU - IF ITYPE.EQ.MENU+N, THE SUBPICTURE REPRESENTS MENU ITEM NUMBER N (STARTING FROM 0)

FUNCTN - THE FUNCTION MESSAGE (ITEM IN TOP LEFT CORNER OF SCREEN TELLING THE USER THE LAST FUNCTION BUTTON HIT).

NOTE: THE LATEST DEFINITIONS ARE IN THE INSERT FILE 'GP/1.INSERT'

XVAL - AN X VALUE IN INCHES (+ OR - 4.5 INCHES)
YVAL - A Y VALUE IN INCHES (+ OR - 4.5 INCHES)
IHEADR - THE HEADER VALUE FOR A POINT, LINE, ETC.

SOME OF THE ABBREVIATIONS USED IN NAMING SUBROUTINES AND VARIABLES ARE:

PIC - SUBPICTURE
PNT - POINT
LIN - LINE
CIR - CIRCLE
CH - CHAIN
CHV - CHAIN VALUE (LINK VALUE)
AR - AREA
ARV - AREA CODE VALUE
SCN - SCAN
QUE - QUEUE
TB - TABLE
DSP - DISPLAY
PUT - PUT SOMETHING INTO A TABLE OR TABLES
GET - GET SOMETHING FROM A TABLE OR TABLES
INIT - INITIALIZE
DEL - DELETE SOMETHING FROM A TABLE OR TABLES
LP - LIGHT PEN
IN - INPUT
MSG - MESSAGE AREA AT TOP OF SCREEN
ACK - ACKNOWLEDGEMENT AREA AT TOP OF SCREEN
MNU - MENU AREA AT BOTTOM OF SCREEN
FUN - FUNCTION
ERR - ERROR
OT - OUTPUT
TEMP - TEMPORARY
SUB - SUBPICTURE
AK - ARROW KEY
KEY - FUNCTION KEY

THE VARIABLE NAMING CONVENTION IS STANDARD FORTRAN.

TO PUT NUMERICAL DATA IN DISPLAY FIELDS, SEE THE PRIME 'ENCODE' FUNCTION.

THE SUBROUTINES ARE DEFINED AS FOLLOWS:

TINIT - TABLE INITIALIZATION - ALL TABLES ARE RESET. USED BY THE MAIN PROGRAM. (SUBROUTINE NUMBER 5.2)
SPECIFY: NONE
RETURNED: NONE

```
        ERRORS TRAPPED: NONE

SINIT  - SCREEN INITIALIZATION SUBROUTINE. THIS SUB IS CALLED BY THE
        MAIN PROGRAM. IT PUTS UP THE BACKGROUND FOR THE SCREEN AND THE
        FUNCTION KEYS ALONG THE SIDE. (SUBROUTINE NUMBER 5.1)
        SPECIFY: NONE
        RETURNED: NONE
        ERRORS TRAPPED: NONE

SELECT(ICALL,NSUB,ITYPE) - ENABLE LIGHTPEN HIT ON SUBPICTURE AND
        RETURN SUBPICTURE AND TYPE. (SUBROUTINE NUMBER 7)
        SPECIFY: ICALL - CALL NUMBER
        RETURNED: NSUB - SUBPICTURE NUMBER
                  ITYPE - SUBPICTURE TYPE
        ERRORS TRAPPED: NONE

SELCT2(ICALL,NSUB,ITYPE,XVAL,YVAL) - ENABLE LIGHTPEN HIT ON SUBPICTURE AND
        RETURN SUBPICTURE, LOC β TYPE. (SUBROUTINE NUMBER 7.1)
        NOTE: (XVAL,YVAL) IS THE LOCATION OF THE ORIGIN OF THE SUBPICTURE
        CALL, NOT THE LOCATION OF THE LIGHTPEN HIT ON THE SCREEN.
        SPECIFY: ICALL - CALL NUMBER
        RETURNED: NSUB - SUBPICTURE NUMBER
                  ITYPE - SUBPICTURE TYPE
                  XVAL - X VALUE OF HIT (CURRENTLY LOC OF SUB-SUBPIC)
                  YVAL - Y VALUE OF HIT (CURRENTLY LOC OF SUB-SUBPIC)
        ERRORS TRAPPED: NONE

PUTPNT(ICALL,NSUB,IHEADR,XVAL,YVAL) - PUT A POINT INTO THE SUBPICTURE
        TABLE AND POINT TABLE, AND DISPLAY. (SUBROUTINE 11)
        SPECIFY: ICALL - CALL NUMBER
                 IHEADR - HEADER VALUE
                 XVAL - X VALUE OF POINT
                 YVAL - Y VALUE OF POINT
        RETURNED: NSUB - SUBPICTURE NUMBER POINT PUT INTO
        ERRORS TRAPPED: SUBPICTURE NOT AVAILABLE
                        POINT TABLE OVERFLOW

PUTPT3(ICALL,NSUB,IHEADR,XVAL,YVAL,ZVAL) - PUT A POINT INTO THE SUBPICTURE
        TABLE AND POINT TABLE, AND DISPLAY (3D). (SUBROUTINE 11.1)
        SPECIFY: ICALL - CALL NUMBER
                 IHEADR - HEADER VALUE
                 XVAL - X VALUE OF POINT
                 YVAL - Y VALUE OF POINT
                 ZVAL - Z VALUE O POINT
        RETURNED: NSUB - SUBPICTURE NUMBER POINT PUT INTO
        ERRORS TRAPPED: SUBPICTURE NOT AVAILABLE
                        POINT TABLE OVERFLOW

GETPNT(ICALL,NSUB,IHEADR,XVAL,YVAL) - READ A POINT FROM TABLES
        (SUROUTINE 10)
```

SPECIFY: ICALL - CALL NUMBER
NSUB - SUBPICTURE NUMBER
RETURNED: IHEADR - HEADER VALUE
XVAL - X VALUE
YVAL - Y VALUE
ERRORS TRAPPED: SUBPICTURE NOT POINT
SUBPICTURE NOT DEFINED

GETPT3(ICALL,NSUB,IHEADR,XVAL,YVAL,ZVAL) - READ A POINT FROM TABLES (3D)
(SUROUTINE 10.1)
SPECIFY: ICALL - CALL NUMBER
NSUB - SUBPICTURE NUMBER
RETURNED: IHEADR - HEADER VALUE
XVAL - X VALUE
YVAL - Y VALUE
ZVAL - Z VALUE
ERRORS TRAPPED: SUBPICTURE NOT POINT
SUBPICTURE NOT DEFINED

PUTLIN(ICALL,NSUB,IHEADR,X1,Y1,X2,Y2,TVAL) - SUBROUTINE TO PUT A LINE INTO
THE TABLES, AND DISPLAY. (SUBROUTINE 17)
SPECIFY: IHEADR - HEADER VALUE
X1 - X VALUE OF FIRST POINT
Y1 - Y VALUE OF FIRST POINT
X2 - X VALUE OF SECOND POINT
Y2 - Y VALUE OF SECOND POINT
TVAL - T VALUE FOR THE LINE (USED FOR A,B,C,S β T FORMAT)
ICALL - CALL NUMBER
RETURNED: NSUB - SUBPICTURE NUMBER
ERRORS TRAPPED: SUBPICTURE NOT AVAILABLE
LINE TABLE OVERFLOW

PUTLN3(ICALL,NSUB,IHEADR,X1,Y1,Z1,X2,Y2,Z2) - SUBROUTINE TO PUT A LINE INTO
THE TABLES, AND DISPLAY (3D). (SUBROUTINE 17.1)
SPECIFY: IHEADR - HEADER VALUE
X1 - X VALUE OF FIRST POINT
Y1 - Y VALUE OF FIRST POINT
Z1 - Z VALUE OF FIRST POINT
X2 - X VALUE OF SECOND POINT
Y2 - Y VALUE OF SECOND POINT
Z2 - Z VALUE FOR SECOND POINT
ICALL - CALL NUMBER
RETURNED: NSUB - SUBPICTURE NUMBER
ERRORS TRAPPED: SUBPICTURE NOT AVAILABLE
LINE TABLE OVERFLOW

GETLIN(ICALL,NSUB,IHEADR,X1,Y1,X2,Y2,TVAL) - READ A LINE FROM THE TABLES
(SUBROUTINE 16)
SPECIFY: ICALL - CALL NUMBER
NSUB - SUBPICTURE NUMBER

```
          RETURNED: IHEADR - HEADER VALUE
                    X1 - X VALUE OF FIRST POINT
                    Y1 - Y VALUE OF FIRST POINT
                    X2 - X VALUE OF SECOND POINT
                    TVAL - T VALUE OF LINE
                    Y2 - Y VALUE OF SECOND POINT
          ERRORS TRAPPED: SUBPICTURE NOT LINE
                          SUBPICTURE NOT DEFINED

GETLN3(ICALL,NSUB,IHEADR,X1,Y1,Z1,X2,Y2,Z2) - READ A LINE FROM 3-D TABLES
          (SUBROUTINE 16.1)
          SPECIFY: ICALL - CALL NUMBER
                   NSUB - SUBPICTURE NUMBER
          RETURNED: IHEADR - HEADER VALUE
                    X1 - X VALUE OF FIRST POINT
                    Y1 - Y VALUE OF FIRST POINT
                    Z1 - Z VALUE OF FIRST POINT
                    X2 - X VALUE OF SECOND POINT
                    Y2 - Y VALUE OF SECOND POINT
                    Z2 - Z VALUE OF SECOND POINT
          ERRORS TRAPPED: SUBPICTURE NOT LINE
                          SUBPICTURE NOT DEFINED

PUTCIR(ICALL,NSUB,IHEADR,XVAL,YVAL,RADIUS,BEGANG,ENDANG) - PUT CIRCLE OR
          ARC INTO THE TABLES, AND DISPLAY. (SUBROUTINE 19)
          SPECIFY: ICALL - CALL NUMBER
                   IHEADT - HEADER VALUE
                   XVAL - X VALUE OF THE CENTER
                   YVAL - Y VALUE OF THE CENTER
                   RADIUS - RADIUS OF THE CIRCLE OR ARC (INCHES)
                   BEGANG - BEGINNING ANGLE OF ARC (RADIANS)
                   ENDANG - ENDING ANGLE OF ARC (RADIANS) NOTE: ARC
                            DRAWN COUNTERCLOCKWISE FOR POS ANGLE.
          RETURNED: NSUB - SUBPICTURE NUMBER
          ERRORS TRAPPED: SUBPICTURE NOT AVAILABLE
                          CIRCLE/ARC TABLE OVERFLOW

GETCIR(ICALL,NSUB,IHEADR,XVAL,YVAL,RADIUS,BEGANG,ENDANG) - READ CIRCLE OR
          ARC FROM TABLES. (SUBROUTINE 18)
          SPECIFY: ICALL - CALL NUMBER
                   NSUB - SUBPICTURE NUMBER
          RETURNED: IHEADT - HEADER VALUE
                    XVAL - X VALUE OF THE CENTER
                    YVAL - Y VALUE OF THE CENTER
                    RADIUS - RADIUS OF THE CIRCLE OR ARC (INCHES)
                    BEGANG - BEGINNING ANGLE FOR ARC (RADIANS)
                    ENDANG - ENDING ANGLE FOR ARC (RADIANS) NOTE: THE
                             ARC IS DRAWN COUNTERCLOCKWISE FOR POS ANGLE.
          ERRORS TRAPPED: SUBPICTURE NOT CIRCLE OR ARC
                          SUBPICTURE NOT DEFINED
```

```
CHIN(ICALL,NSUB,IHEADR) - ACCEPTS LIGHTPEN INPUT OF CHAIN FROM THE USER
          AND PLACES A CHAIN INTO THE CHAIN TABLES (SUBROUTINE 20).
          SPECIFY: ICALL - CALL NUMBER
                   IHEADT - HEADER VALUE
          RETURNED: NSUB - SUBPICTURE NUMBER
          ERRORS TRAPPED: CHAIN TABLE OVERFLOW
                          CHAIN LINK VALUE TABLE OVERFLOW

DSPCH(NCHAIN) - DISPLAYS A CHAIN ON THE SCREEN (SUBROUTINE 31)
          SPECIFY: NCHAIN - CHAIN NUMBER
          RETURNED: (NONE)
          ERRORS TRAPPED: INVALID CHAIN

DRLINK(ICALL,IXROOT,IYROOT,LINK) - DRAWS A LINK IN A CHAIN (USED BY
          SUBROUTINE CHIN AND DSPCH).(SUBROUTINE 21)
          SPECIFY: ICALL - CALL NUMBER
                   IXROOT - X VALUE (LEVEL 0) OF ROOT OF LINK
                   IYROOT - Y VALUE (LEVEL 0) OF ROOT OF LINK
                   LINK - VALUE OF LINK (0 TO 7)
          RETURNED: (NONE)
          ERRORS TRAPPED: BAD LINK VALUE

SCAN(ICALL,NCHAIN,IOPT) - SCANS A CHAIN AND FILLS THE SCAN TABLE (ISCAN)
          WITH EDGE VALUES (SUBROUTINE 22).
          SPECIFY: ICALL - CALL NUMBER
                   NCHAIN - NUMBER OF THE CHAIN
                   IOPT - OPTION VALUE (ISCAN IS CLEARED IF IOPT IS 0 OR 1,
                   NOT CLEARED IF 2 OR LARGER)
          ERRORS TRAPPED: LEVEL MISMATCH
                          BAD LINK VALUE ENCOUNTERED
                          SCAN TABLE OVERFLOW

SORT - SORTS THE SCAN TABLE IN INCREASING ORDER FIRST BY Y AND THEN
          BY X (WITHIN VALUES OF Y). REMOVES REDUNDANT VALUES.
          (SUBROUTINE 23)
          SPECIFY: (NONE)
          RETURNED: (NONE)
          ERRORS TRAPPED: (NONE)

QINS(IX,IY,IERR) - INSERT AN ELEMENT (X,Y) IN THE QUEUE (SUBROUTINE 27).
          SPECIFY: IX - X VALUE
                   IY - Y VALUE
          RETURNED: IERR - ERROR CODE (NO ERROR IF 0, QUEUE OVERFLOW IF 1)
          ERRORS TRAPPED: QUEUE OVERFLOW

QINSD(IX,IY,IERR) - SAME AS QINS BUT FOR I*32 INTEGERS (USED BEFORE
          PROGRAM WAS CONVERTED TO ALL I*32). (SUBROUTINE 38)

QDEL(IX,IY,IERR) - DELETE ELEMENT (X,Y) FROM QUEUE. (SUBROUTINE 28)
```

SPECIFY: IX - X VALUE
IY - Y VALUE
RETURNED: IERR - ERROR CODE (NO ERROR IF 0, QUEUE UNDERFLOW IF 1)
ERRORS TRAPPED: UNDERFLOW

QDELD(IX,IY,IERR) - SAME AS QDEL BUT FOR I*32 INTEGERS (BEFORE PROGRAM WAS CONVERTED TO ALL I*32). (SUBROUTINE 39)

GETSUB(ICALL,NSUB,ITYPE) - GET AN AVAILABLE (NOT USED) SUBPICTURE. (SUBROUTINE 12)
SPECIFY: ICALL - CALL NUMBER
ITYPE - THE TYPE OF THE SUBPICTURE BEING DEFINED (200 TO 299 IS RESERVED FOR TEMPORARY USE). NOTE: TYPE 0 SHOULD NEVER BE USED (IT INDICATES THAT THE SUBPICTURE IS NOT BEING USED).
RETURNED: NSUB - SUBPICTURE NUMBER
ERRORS TRAPPED: NO SUBPICTURE AVAILABLE
ITYPE OF 0 SPECIFIED .

DELSUB(ICALL,NSUB) - DELETE SUBPICTURE - THE INDICATED ELEMENT IS DELETED AND CLEARED. THE SUBPICTURE IS MADE AVAILABLE AND THE DISPLAY OF THE ELEMENT IS ERASED FROM THE SCREEN. (SUBROUTINE 6)
SPECIFY: ICALL - CALL NUMBER
NSUB - SUBPICTURE NUMBER
RETURNED: NONE
ERRORS TRAPPED: NSUB OUT OF RANGE

SEARCH(NSUB,ITYPE,ISTART) - SEARCH THE SUBPICTURE TABLE FOR A SUBPICTURE OF TYPE 'ITYPE' BEGINNING WITH SUBPICTURE 'ISTART'. ALL OF THE ELEMENTS OF A SPECIFIC KIND CAN BE FOUND BY STEPPING ISTART FROM 1 TO 'MAXPIC'. (SUBROUTINE NUMBER 8)
SPECIFY: ITYPE - SUBPICTURE TYPE
ISTART - STARTING SUBPICTURE TO BEGIN SEARCH (1 TO 'MAXPIC')
RETURNED: NSUB - SUBPICTURE NUMBER (0 IF NONE FOUND)
ERRORS TRAPPED: NONE

LPIN(XVAL,YVAL) - LIGHTPEN INPUT OF COORDINATES OF A POINT ON THE SCREEN. WHEN CALLED, THE TRACKING CROSS IS PUT UP AND CAN BE MOVED BY THE LIGHTPEN. WHEN CONFIRMED BY FK-0, THE X AND Y VALUES ARE RETURNED TO THE CALLING PROGRAM. THE TRACKING CROSS IS PUT UP AT THE CENTER OF THE SCREEN. (SUBROUTINE 13)
SPECIFY: NONE
RETURNED: XVAL - X VALUE OF THE INDICATED POINT
YVAL - Y VALUE OF THE INDICATED POINT
ERRORS TRAPPED: NONE

LPIN2(XVAL,YVAL) - SAME AS LPIN EXCEPT INITIALIZES CURSOR AT (XVAL,YVAL) INSTEAD OF AT (0,0). (SUBROUTINE 13.1)

```
LPIN3(IOPT,XVAL,YVAL,IEND) - ENABLES THE LIGHTPEN (IOPT = 0 OR 1), READS A
            POINT IMMEDIATELY (WITHOUT CONFIRM) (IOPT = 2) OR DISABLES THE
            LIGHTPEN
            (IOPT = 3). (SUBROUTINE 13.2)
            THE ARGUMENTS ARE AS FOLLOWS:
              IOPT - THE DESIRED OPTION
                       0 - ENABLE LP AT CURRENT LOCATION
                       1 - ENABLE LP AT (XVAL,YVAL) (NO VALUES ARE TAKEN)
                       2 - READ THE LOCATION OF THE TRACKING CROSS INTO
                           (XVAL,YVAL) WITHOUT WAITING FOR USER CONFIRM (USER
                           HAS HIT FK-0 IF IEND=1, IEND=0 OTHERWISE)
                       3 - DISABLES LP (REQUIRED!!!!!!!!!!!!)
              (XVAL,YVAL) - LOCATION FOR ENABLE (IOPT=1) OR RETURNED
                            LOCATION (IOPT=2)
              IEND - END PARAMETER (IOPT=2 ONLY) (RETURNED)
                       1 - USER HAS HIT FK-0
                       0 - OTHERWISE
            NOTE: IOPT IS NOT MODIFIED BY THIS SUBROUTINE. IT CAN BE
                  SPECIFIED BY A LITERAL VALUE. IEND IS NOT MODIFIED FOR
                  IOPT = 0, 1, OR 4. IOPT = 2, IT IS MODIFIED. DO NOT SPECIFY
                  IEND WITH A LITERAL FOR IOPT =2!!!

AKIN(XVAL,YVAL) - ARROW KEY INPUT OF COORDINATES OF A POINT ON THE SCREEN.
           WHEN CALLED, THE CROSSHAIR IS PUT UP AND CAN BE MOVED
           BY THE ARROW KEYS. WHEN CONFIRMED BY FK-0, THE X AND Y VALUES
           ARE RETURNED TO THE CALLING PROGRAM. THE CROSSHAIR IS PUT
           UP AT THE CENTER OF THE SCREEN. (SUBROUTINE 13.3)
           SPECIFY: NONE
           RETURNED: XVAL - X VALUE OF THE INDICATED POINT
                     YVAL - Y VALUE OF THE INDICATED POINT
           ERRORS TRAPPED: NONE

 AKIN2(XVAL,YVAL) - SAME AS AKIN EXCEPT INITIALIZES CURSOR AT (XVAL,YVAL)
           INSTEAD OF AT (0,0). (SUBROUTINE 13.4)

 AKIN3(IOPT,XVAL,YVAL,IEND) - ENABLES THE CROSSHAIR (IOPT = 0 OR 1), READS A
            POINT IMMEDIATELY (WITHOUT CONFIRM) (IOPT = 2) OR DISABLES THE
            CROSSHAIR (IOPT = 3). (SUBROUTINE 13.5)
            THE ARGUMENTS ARE AS FOLLOWS:
              IOPT - THE DESIRED OPTION
                       0 - ENABLE AK AT CURRENT LOCATION
                       1 - ENABLE AK AT (XVAL,YVAL) (NO VALUES ARE TAKEN)
                       2 - READ LOCATION OF THE CROSSHAIR INTO (XVAL,YVAL)
                           WITHOUT WAITING FOR USER CONFIRM (USER HAS HIT
                           FK-0 IF IEND=1, IEND=0 OTHERWISE)
                       3 - DISABLES AK (REQUIRED!!!!!!!!!!!!)
              (XVAL,YVAL) - LOCATION FOR ENABLE (IOPT=1) OR RETURNED LOCATION
                            (IOPT=2)
              IEND - END PARAMETER (IOPT=2 ONLY) (RETURNED)
                       1 - USER HAS HIT FK-0
```

```
                    0 - OTHERWISE
          NOTE: IOPT NOT MODIFIED BY THIS SUBROUTINE. IT CAN BE SPECIFIED
                BY A LITERAL VALUE. IEND IS NOT MODIFIED FOR IOPT = 0, 1,
                OR 4. FOR IOPT = 2, IT IS MODIFIED. DO NOT SPECIFY IEND
                WITH A LITERAL FOR IOPT =2!!!

GETPC(I,IREADY,NLEV,IDS,XYLOCS) - GET SUBPICTURE - MODIFIED GETPIC WHICH
          DOES NOT REQUIRE CONFIRMATION OF PICK. (SUBROUTINE 14)
          (PARAMETERS ARE IDENTICAL TO GETPIC)
          ERRORS TRAPED: NONE

DSPMSG('(TEXT)',NCHAR) - ADD A MESSAGE TO BE DISPLAYED. THE NEW
          ITEM IS ADDED TO THE RIGHT OF THE EXISTING MESSAGE ITEMS.
          THE ITYPE IS INCREMENTED AND THE SUBPICTURE IS LIGHTPEN
          SENSITIVE.
          IF NCHAR IS 0, THE MESSAGE FIELD IS CLEARED AND THE ITEMS ENDED.
          THERE IS CURRENTLY  A MAXIMUM OF 15 ITEMS. (SUBROUTINE 9.2)
          SPECIFY: '(TEXT') - TEXT OF MESSAGE (TO PUT IN VARIABLES, SEE
                             PRIME FORTRAN FUNCTION 'ENCODE')
                   NCHAR - NUMBER OF CHARACTERS IN THE MESSAGE (TOTAL OF 71)
          RETURNED: (NONE)
          ERRORS TRAPPED: NONE

DSPMG2(IIOPT,'(TEST)',NCHAR) - SAME AS DSPMSG EXCEPT MESSAGE ITEMS ARE
           DISPLAYED WITH TWO POSSIBLE INTENSITIES. THIS IS TYPICALLY
           USED TO INDICATE THAT AN OPTION IS CURRENTLY IN EFFECT (IF
           AT THE HIGH INTENSITY). (SUBROUTINE 9.3)
           SPECIFY: IIOPT - IF 0, DISPLAY AT NORMAL INTENSITY
                            IF NOT 0, DISPLAY AT MAXIMUM INTENSITY

DSPMNU('(TEXT)',NCHAR) - ADD A MENU ITEM TO BE DISPLAYED. THE NEW
          ITEM IS ADDED TO THE RIGHT OF THE EXISTING ITEMS (UP TO
          A MAXIMUM OF 15). EACH IS PUT IN A NEW SUBPICTURE AND EACH IS
           LIGHTPEN SENSITIVE. IF ICHAR IS 0, ALL EXISTING MENU ITEMS ARE
           ERASED AND THE NUMBERS RESET. (SUBROUTINE 9.1)
           SPECIFY: '(TEXT)' - TEXT OF THE MENU ITEM. NOTE: TO PUT IN A
                               VARIABLE, SEE THE PRIME FORTRAN FUNCTION
                               'ENCODE'.
                    NCHAR - NUMBER OF CHARACTERS (TOTAL OF 66)
           RETURNED: (NONE)
           ERRORS TRAPPED: NONE

DSPFUN(IFUN) - DISPLAY FUNCTION ON SCREEN - THE FUNCTION SPECIFIED IS
           DISPLAYED AT THE TOP LEFT OF THE SCREEN. THE TEXT IS IDENTICAL
           TO THAT DISPLAYED BELOW THE FUNCTION NUMBER AT THE EDGES OF
           THE SCREEN. IT IS CLEARED IF IFUN IS LESS THAN 0. IFUN IS
           THE FUNCTION FUNKEY KEY NUMBER (0 TO 31). (SUBROUTINE 9.4)
           SPECIFY: IFUN - DESIRED FUNCTION NUMBER (0 TO 31)
           RETURNED: NONE
           ERRORS TRAPPED: ILLEGAL NUMBER
```

```
DSPERR('(TEXT)',NCHAR) - DISPLAY A FLASHING ERROR MESSAGE. THE
          EXISTING ERROR MESSAGE (IF ANY) IS CLEARED AND THE NEW MESSAGE
          IS PUT UP RIGHT JUSTIFIED (SLID ALL THE WAY TO THE RIGHT). IF
          NCHAR IS 0, THE EXISTING MESSAGE IS DELETED.
          (SUBROUTINE 9.5)
          SPECIFY: '(TEXT)' - TEXT OF ERROR
                   NCHAR - NUMBER OF CHARACTERS IN ERROR MESSAGE
                           (UP TO 57)
          RETURNED: NONE
          ERRORS TRAPPED: NONE

DSPACK('(TEXT)',NCHAR) - DISPLAY ACKNOWLEDGEMENT - ANY OLD
          MESSAGE IS CLEARED AND THE NEW MESSAGE IS DISPLAYED. IF NCHAR
          IS 0, ANY EXISTING ITEM IS DELETED. (SUBROUTINE 15)
          SPECIFY: '(TEXT)' - TEXT OF ACKNOWLEDGEMENT
                   NCHAR - NUMBER OF CHARACTERS (UP TO 23)
          RETURNED: NONE
          ERRORS TRAPPED: NONE

EXIST(FILE,ISTAT) - THE EXISTENCE OF A FILE IS DETERMINED. (SUBROUTINE 46)
          THE ARGUMENTS ARE AS FOLLOWS:
            FILE - FILE NAME (UP TO 32 CHARACTERS)
            ISTAT - STATUS OF FILE (RETURNED)
                      0 - FILE EXISTS
                     OTHER - CORRESPONDS TO PRIME ERROR CODE

  NOTE: THE LENGTH OF THE NAME IS DETERMINED BY SEARCHING FOR THE FIRST
        BLANK IN THE FILE NAME. IF NONE IS FOUND, 32 IS ASSUMED. IN ORDER TO
        MAINTAIN THIS CONVENTION, DEFINE THE NAME ('FILE') AS FOLLOWS:
                   INTEGER*4 FILE(8)
        THE READ STATEMENT SHOULD BE OF THE FORM:
                   READ(1,10) FILE
             10    FORMAT(8A4)

        THIS IS TRUE FOR ALL THE FILE NAMES USED BELOW.

EXIST2(NAME1,NAME2,ISTAT) - THE STATUS OF A FILE IS DETERMINED. THE NAME OF
          THE FILE IS 'NAME1.NAME2'. (SUBROUTINE 46.1)
          THE ARGUMENTS ARE AS FOLLOWS:
            NAME1 - FIRST HALF OF THE FILE NAME (UP TO 31 CHARACTERS)
            NAME2 - SECOND HALF OF FILE NAME (UP TO 31 CHARACTERS).
                    THE FILE NAME WILL BE THE CONCATENATION OF THE TWO
                    PARTS WITH A PERIOD IN THE CENTER ('NAME1.NAME2').
            ISTAT - STATUS OF FILE (RETURNED)
                      0 - FILE EXISTS
                     OTHER - CORRESPONDS TO PRIME ERROR CODE

OPEN(IOPT,IFUNIT,FILE) - OPEN A FILE ON A SPECIFIED FORTRAN UNIT
                         (SUBROUTINE 47)
```

```
            THE ARGUMENTS ARE AS FOLLOWS:
              IOPT - OPTION SELECTION FOR READ OR WRITE
                        1 - READ
                        2 - WRITE
                        3 - READ OR WRITE
              IFUNIT - FORTRAN UNIT (PRIMOS UNIT - 4)
              FILE - FILE NAME

OPEN2(IOPT,IFUNIT,NAME1,NAME2) - OPEN A FILE ON FORTRAN UNIT (TWO PART FILE
           NAME) (SUBROUTINE 47.1)
           THE ARGUMENTS ARE AS FOLLOWS:
             IOPT - OPTION SELECTION FOR READ OR WRITE
                        1 - READ
                        2 - WRITE
                        3 - READ OR WRITE
             IFUNIT - FORTRAN UNIT NUMBER (PRIMOS UNIT - 4)
             NAME1 - FIRST HALF OF FILE NAME
             NAME2 - SECOND HALF OF FILE NAME (WILL BE 'NAME1.NAME2')

CLOSE(IFUNIT) - THE FILE OPEN ON FORTRAN UNIT IFUNIT IS CLOSED (PRIMOS
                UNIT - 4) (SUBROUTINE 48)
           THE ARGUMENT IS AS FOLLOWS:
             IFUNIT - FORTRAN UNIT NO (PRIMOS - 4)

READ(IFUNIT,IRECNO,IBUFF,LENREC,IERR) - READ A BUFFER FROM AN OPEN
           DISC FILE. IF IRECNO=0, READ AT
           CURRENT DISC LOCATION. IF NOT, START AT NUMBERED RECORD. RECORDS
           ARE LENREC WORDS (16 BITS) IN LENGTH. RECORDS NUMBERED FROM 1.
           (SUBROUTINE 49)
           THE ARGUMENTS ARE AS FOLLOWS:
               IFUNIT - FORTRAN UNIT NUMBER
               IRECNO - RECORD NUMBER FROM WHICH TO READ (RECORDS OF LENGTH
                        LENREC WORDS (16 BITS) NUMBERED FROM 1). IF
                        IRECNO=0, THE NEXT RECORD IS READ.
               IBUFF - BUFFER AREA (FIRST WORD OF ARRAY)
               LENREC - LENGTH OF RECORD IN WORDS (16 BITS EACH)
                        NOTE: CAN BE ANY LENGTH (NOT RESTRICTED TO 64K).
               IERR - ERROR CODE (0 IF NO ERROR, PRIMOS CODE OTHERWISE).
                      NOTE: AN ERROR MESSAGE WILL BE GENERATED IF IERR.NE.0.

WRITE(IFUNIT,IRECNO,IBUFF,LENREC,IERR) - WRITE A BUFFER INTO
           AN OPEN DISC FILE. IF IRECNO=0, WRITE AT
           CURRENT DISC LOCATION. IF NOT, START AT NUMBERED RECORD. RECORDS
           ARE LENREC WORDS (16 BITS) IN LENGTH. RECORDS NUMBERED FROM 1.
           (SUBROUTINE 50)
           THE ARGUMENTS ARE AS FOLLOWS:
               IFUNIT - FORTRAN UNIT NUMBER
               IRECNO - RECORD NUMBER INTO WHICH TO WRITE (RECORDS OF LENGTH
                        LENREC WORDS (16 BITS) NUMBERED FROM 1). IF
                        IRECNO=0, THE NEXT RECORD IS WRITTEN.
```

```
                IBUFF - BUFFER AREA (FIRST WORD OF ARRAY)
                LENREC - LENGTH OF RECORD IN WORDS (16 BITS EACH)
                         NOTE: LENGTH CAN BE ANY VALUE (NOT RESTRICTED TO
                               64K).
                IERR - ERROR CODE (0 IF NO ERROR, PRIMOS CODE OTHERWISE).
                       NOTE: AN ERROR MESSAGE WILL BE GENERATED IF IERR.NE.0.

DELETE(FILE) - DELETE NAMED FILE (SUBROUTINE 51)
               THE ARGUMENT IS AS FOLLOWS:
                 FILE - NAME OF FILE

DELET2(NAME1,NAME2) - DELETE NAMED FILE (TWO NAME FORMAT) (SUBROUTINE 51.1)
               THE ARGUMENTS ARE AS FOLLOWS:
                 NAME1 - FIRST PART OF FILE NAME
                 NAME2 - SECOND PART OF FILE NAME (THE FILE NAME WILL BE
                         'NAME1.NAME2')

ATTACH(NAMUFD,IERR) - THIS SUBROUTINE WILL ATTACH TO A UFD. IT ASSUMES THAT
           THERE IS NO PASSWORD. ERROR MESSAGE GENERATED IF UNSUCCESSFUL.
           (SUBROUTINE 52)
           THE ARGUMENTS ARE AS FOLLOWS:
               NAMUFD - NAME OF THE UFD (UP TO 32 CHARACTERS)
               IERR - ERROR RETURN CODE
                        0 - SUCCESSFUL ATTACH
                      OTHER - SEE SYSCOM>ERRD.F
                                E$NATT (7) - NO UFD ATTACHED
                                E$NTUD (12) - NOT A UFD
                                E$FNTF (15) - NOT FOUND
                                 ETC.

ATCHDN(NAMUFD,IERR) - THIS SUBROUTINE WILL ATTACH DOWN TO A SUB-UFD. IT
           ASSUMES THAT THERE IS NO PASSWORD. AN ERROR MESSAGE IS GENERATED
           IF UNSUCCESSFUL. (SUBROUTINE 52.1)
           THE ARGUMENTS ARE AS FOLLOWS:
             NAMUFD - UFD NAME (SAME FORMAT AS FILE NAMES)
             IERR - 0 IS SUCCESSFUL, SEE SYSCOM>ERRD.F OTHERWISE

TOHOME - THIS SUBROUTINE WILL CAUSE AN ATTACH TO THE HOME UFD. NO ERRORS
           ARE REPORTED OR MESSAGES GENERATED. (SUBROUTINE 52.2)

CONCAT(NAME1,NAME2,FILE) - THIS SUBROUTINE WILL CONCATENATE TWO NAMES
           WITH A '.' IN THE MIDDLE. (SUBROUTINE 53)
           THE ARGUMENTS ARE AS FOLLOWS:
               NAME1 - FIRST NAME (UP TO 32 CHARACTERS, 31 CAN BE USED)
               NAME2 - SECOND NAME (UP TO 32 CHARACTERS, 31 CAN BE USED)
               FILE - NEW NAME. HAS A VALUE OF 'NAME1.NAME2'

LINCOF(X1,Y1,X2,Y2,A,B,C) - GENERATE THE A, B AND C COEFFICIENTS FOR THE
           AX+BY+C=0 EQUATION OF THE LINE SPECIFIED BY THE END POINTS
           (X1,Y1) AND (X2,Y2). THE COEFFICIENTS ARE SCALED SO A**2+B**2=1.
```

FOR A VERTICAL LINE B=0. THE SLOPE OF THE LINE IS -A/B.
(SUBROUTINE 33)

INTLL(A1,B1,C1,A2,B2,C2,X,Y) - THE INTERSECTION (X,Y) OF TWO LINES (A1X+B1Y +C1=0) AND A2X+B2Y+C2=0) IS RETURNED. AN ERROR MESSAGE IS GENERATED IF THE LINES ARE PARALLEL (A1*B2 = A2*B1) AND (0,0) IS RETURNED (SUBROUTINE 33.2)

INTERS(X,Y,A,B,C,XI,YI) - THE INTERSECTION OF A LINE (AX+BY+C=0) AND THE NORMAL WHICH PASSES THROUGH A POINT (X,Y) IS RETURNED IN (XI,YI). (SUBROUTINE 33.3)

ANGLL(A1,B1,C1,A2,B2,C2,ANG,ICODE) - THE ANGLE BETWEEN LINE 1 (A1X+B1Y+C1=0) AND LINE 2 (A2X+B2Y+C2=0) IS RETURNED IN ANG. (SUBROUTINE 33.1) THE RETURN CODE (ICODE) IS RETURNED AS FOLLOWS:

1 - LINES PARALLEL
2 - LINES PERPENDICULAR
3 - OTHERWISE

ARCCOS(COS,ANG) - THE INVERSE COSINE OF COS IS RETURNED IN ANG. (SUBROUTINE 32)

ARCSIN(SIN,ANG) - THE INVERSE SINE OF SIN IS RETURNED IN ANG. (SUBROUTINE 32.1)

FUNCTIONS:

FUN0 - POINT INPUT FUNCTION (SUBROUTINE 24)
FUN1 - LINE INPUT FUNCTION (SUBROUTINE 25)
FUN2 - CIRCLE/ARC FUNCTION (SUBROUTINE 26)
FUN6 - VIEWSPACE DEFINITION FUNCTION (SUBROUTINE 30)
FUN7 - DISPLAY OF CHAIN FUNCTION (SUBROUTINE 29)
FUN3 - ERASE A SUBPICTURE (SUBROUTINE 40)
FUN4 - DELETE AN ITEM (SUBROUTINE 41)
FUN5 - CHANGE DEFAULT PARAMETERS (SUBROUTINE 42)
FUN100 - ENTER CHAIN ENCODED BLOB (SUBROUTINE 43)
FUN101 - CHANGE OF GRID LEVEL (SUBROUTINE 44)
FUN102 - DRAW GRID (SUBROUTINE 45)

FUN201 168 GP/1 EXAMPLE - IDENTIFY ITEM

SUBROUTINE FUN201

FUN202 169 LINE &CIRCLE IN VIEWSPACE (NO. 1) SUBPICTURE

SUBROUTINE FUN202

FUN340 170 ROTATE A LINE

SUBROUTINE FUN340
THIS SUBROUTINE WILL ALLOW THE USER TO SELECT AN EXISTING LINE TO BE ROTATED WITH THE LIGHTPEN. IT THEN ENABLES THE LP TRACKING CROSS FOR ANGLE ENTRY. THE ANGLE ENTERED IS THE ANGLE BETWEEN THE X AXIS & THE LINE FROM THE CENTER OF THE WORKSPACE TO THE ENTERED POINT. THE LINE IS THEN DELETED AND REGENERATED AS A LINE OF THE SAME LENGTH ROTATED ABOUT THE FIRST END OF THE LINE.

THE SUBROUTINE WILL CONTINUE TO REQUEST ANGLES. TO TERMINATE, ENTER A POINT WITHIN THE CIRCLE CENTERED ON THE ORIGIN.

FUN341 171 BEZIER CURVE FUNCTION

SUBROUTINE FUN341
THIS SUBROUTINE ALLOWS THE USER TO ENTER A SERIES OF POINTS WITH THE LIGHTPEN (TERMINATE BY A SECOND FK-0 CONFIRM AT THE SAME LOCATION). IT THEN GENERATES THE LINES BETWEEN THE POINTS AND THE BEZIER CURVE. IT ALSO ALLOWS THE USER TO CHANGE OR ADD POINTS.

FACT 171.1 EVALUATE FACTORAL FUNCTION

DOUBLE PRECISION FUNCTION FACT(K)

FUN342 172 FILLET BETWEEN 2 LINES

SUBROUTINE FUN342
THIS PROGRAM FIRST ALLOWS THE USER TO SELECT TWO EXISTING LINES. IT THEN REQUESTS THE RADIUS OF THE CIRCULAR FILLET AND ENABLES THE TRACKING CROSS FOR A MARKER ENTRY. THE FILLET IS THEN GENERATED FROM THE FIRST LINE TO THE SECOND IN A CLOCKWISE MANNER ON THE SIDE WHERE THE MARKER WAS LOCATED.

FUN343 172 FUNCTION 343 - TEST OF SUBROUTINE TILE

SUBROUTINE FUN343

TILE 174 TESELLATION SUBROUTINE

SUBROUTINE TILE(ICODE,XVERTX,YVERTX,XPTS,YPTS,IPTS,
ORIGX,ORIGY,RIX,RIY,UPX,UPY)
QUADRILATERAL (ICODE=1), HEXAGON TYPE 1 (ICODE=2), HEXAGON TYPE 2 (ICODE=3), AND HEXAGON TYPE 3 (ICODE=4). (SEE SCIENTIFIC AMERICAN, JULY, 1975, PP. 112 TO 117.)

VERTEX POINTS ARE STORED IN XVERTX & YVERTX IN CLOCKWISE ORDER.
FOR HEXAGONS THE STARTING POINT IS A (TYPE 1), A (TYPE 2), AND
B (TYPE 3) (SEE SA ARTICLE).

DSPTIL 174.1 DISPLAY REPEATING TILE PATTERN

```
SUBROUTINE DSPTIL(XPTS,YPTS,IPTS,ORIGX,ORIGY)
```

Appendix H - OCTREE Main Program (OCTREE)

```
C
C  ##############################################################################

C
C
C     MAIN PROGRAM: OCTREE
C
C
C  PROGRAM NUMBER 0 - MAIN PROGRAM FOR OCTREE APPLICATIONS
C
C
C    COPYRIGHT 1979, MEAGER SOFTWARE LIMITED, LOUDONVILLE, NEW YORK
C
$INSERT GP/1.INSERT
C
C  GP/1 - FUNCTION KEY LABELS
C
      DATA FUN/
     0 '  POINT ',     /* LABEL FOR FUNCTION KEY NUMBER  0
     1 '   LINE ',     /* LABEL FOR FUNCTION KEY NUMBER  1
     2 '  CIRCLE',     /* LABEL FOR FUNCTION KEY NUMBER  2
     3 '  CHAIN ',     /* LABEL FOR FUNCTION KEY NUMBER  3
     4 'QUADTREE',     /* LABEL FOR FUNCTION KEY NUMBER  4
     5 ' CIR CH ',     /* LABEL FOR FUNCTION KEY NUMBER  5
     6 ' EXPAND ',     /* LABEL FOR FUNCTION KEY NUMBER  6
     7 ' DISPLAY',     /* LABEL FOR FUNCTION KEY NUMBER  7
     8 '  EDGE  ',     /* LABEL FOR FUNCTION KEY NUMBER  8
     9 '  ERASE ',     /* LABEL FOR FUNCTION KEY NUMBER  9
     0 '        ',     /* LABEL FOR FUNCTION KEY NUMBER 10
     1 ' DELETE ',     /* LABEL FOR FUNCTION KEY NUMBER 11
     2 ' SCAN2D ',     /* LABEL FOR FUNCTION KEY NUMBER 12
     3 '  SLICE ',     /* LABEL FOR FUNCTION KEY NUMBER 13
     4 'ILLUM JS',     /* LABEL FOR FUNCTION KEY NUMBER 14
     5 ' HD COPY',     /* LABEL FOR FUNCTION KEY NUMBER 15 (DON'T CHANGE!)
     6 '  GRID  ',     /* LABEL FOR FUNCTION KEY NUMBER 16
     7 ' LEVEL  ',     /* LABEL FOR FUNCTION KEY NUMBER 17
     8 '  PRINT ',     /* LABEL FOR FUNCTION KEY NUMBER 18
     9 '  MOVE  ',     /* LABEL FOR FUNCTION KEY NUMBER 19
     0 ' ROT 90 ',     /* LABEL FOR FUNCTION KEY NUMBER 20
     1 ' ROTATE ',     /* LABEL FOR FUNCTION KEY NUMBER 21
     2 ' SCALE  ',     /* LABEL FOR FUNCTION KEY NUMBER 22
     3 'FILE.CLD',     /* LABEL FOR FUNCTION KEY NUMBER 23
     4 '   NC   ',     /* LABEL FOR FUNCTION KEY NUMBER 24
     5 'VIEW GS ',     /* LABEL FOR FUNCTION KEY NUMBER 25
     6 ' VIEW3D ',     /* LABEL FOR FUNCTION KEY NUMBER 26
     7 ' BOOLEAN',     /* LABEL FOR FUNCTION KEY NUMBER 27
     8 ' TABLES ',     /* LABEL FOR FUNCTION KEY NUMBER 28
     9 '  FILE  ',     /* LABEL FOR FUNCTION KEY NUMBER 29
     0 ' PARMS  ',     /* LABEL FOR FUNCTION KEY NUMBER 30
     1 '  EXIT  '/     /* LABEL FOR FUNCTION KEY NUMBER 31 (DON'T CHANGE!)
C
```

```
      WRITE(1,5001)
5001  FORMAT('***** ENTER MAXIMUM NUMBER OF OBJECT NODES')
      READ(1,*) MAXARV
C  INITIALIZE TABLES
C
      CALL TINIT
C
C  INITIALIZE SCREEN
C
      CALL GRESET
      CALL ENTGRA
C
CCCCCCC     CALL SINIT
      CALL SVTITL(0)
C
C
C  GET FIRST LP HIT
C
99    CONTINUE
      CALL DSPFUN(-1)
C
       CALL SELECT(5,NSUB,ITYPE)
C
      KEY=ITYPE-FUNKEY
C
C  DECODE - CHECK TO SEE IF NOT FUNKEY FIRST
C
      IF((KEY.LT.0).OR.(KEY.GT.31)) GO TO 99
C
C  PUT KEY HIT IN THE UPPER LEFT HAND CORNER
C
      CALL DSPFUN(KEY)
C
C  CLEAR ACK, ERRORS, MESSAGES β MENU (UNLESS HARDCOPY (KEY 15))
C
      IF(KEY.EQ.15) GO TO 8000
      CALL DSPERR(' ',0)
      CALL DSPACK(' ',0)
      CALL DSPMSG(' ',0)
      CALL DSPMNU(' ',0)
8000  CONTINUE
C
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,
     2 20,21,22,23,24,25,26,27,28,29,30,31), KEY
C
C  KEY 0 GOES HERE (DROPS THROUGH COMPUTED GO TO)
C  NOTE: USE ONLY STATEMENT NUMBERS 32 TO 98 IN HERE!!
C
      CALL FUN0             /* USER ENTRY OF A POINT
      GO TO 99
1     CONTINUE
      CALL FUN1             /* USER ENTRY OF A LINE
      GO TO 99
2     CONTINUE
```

```
      CALL FUN2              /* USER ENTRY OF A CIRCLE OR ARC
      GO TO 99
3     CONTINUE
      CALL FUN100
      GO TO 99
4     CONTINUE
      CALL FUN407  /* CONVERT CHAIN TO AREA
      GO TO 99
5     CONTINUE
      CALL FUN412      /* CIRCLE CHAIN OR SPHERE GENERATION
      GO TO 99
6     CONTINUE
      CALL FUN404  /* EXPAND AN OBJECT THROUGH THE UNIVERSE
      GO TO 99
7     CONTINUE
      CALL FUN7A   /* DISPLAY A CHAIN OR AREA
      GO TO 99
8     CONTINUE
      CALL FUN402  /* DISPLAY EDGES OF AN AREA
      GO TO 99
9     CONTINUE
      CALL FUN3              /* ERASE
      GO TO 99
10    CONTINUE
      GO TO 99
11    CONTINUE
      CALL FUN4                 /* DELETE ITEM
      GO TO 99
12    CONTINUE
      CALL FUN403  /* DISPLAY VISIBLE EDGES OF AN AREA IN SORTED ORDER
      GO TO 99
13    CONTINUE
      CALL FUN411
      GO TO 99
14    CONTINUE
      CALL ILLJS       /* DEANZA ILLUMINATION BY JOYSTICK
      GO TO 99
15    CONTINUE
C
C  *** NOTE: HARDCOPY IS IMPLEMENTED INSIDE SUBROUTINES SELECT β SELCT2.
C            THIS ALLOWS HARDCOPY TO BE GENERATED WHENEVER THE LIGHTPEN
C            IS ENABLED (EVEN IN THE MIDDLE OF A FUNCTION). DO NOT PUT
C            ANY CODE IN HERE (IT WOULD NEVER BE EXECUTED) AND DO NOT
C            CHANGE THE FUNCTION OF KEY 15!
C
      GO TO 99
C
16    CONTINUE
      CALL FUN102                  /* GRID
      GO TO 99
17    CONTINUE
      CALL FUN101
      GO TO 99
18    CONTINUE
```

```
      CALL FUN408  /* GIVE INFO ABOUT CHAINS β AREAS
      GO TO 99
19    CONTINUE
      CALL FUN409  /* TRANSLATE (MOVE) AN AREA
      GO TO 99
20    CONTINUE
      CALL FUN400  /* ROTATE AN AREA BY 90 DEG.
      GO TO 99
21    CONTINUE
      CALL FUN401  /* ROTATE AN AREA BY 0 TO PI/2
      GO TO 99
22    CONTINUE
      CALL FUN413         /* SCALE AN OBJECT
      GO TO 99
23    CONTINUE
      CALL FUN417          /* SAVE/RESTORE IN OLD FORMAT
      GO TO 99
24    CONTINUE
      CALL FUN414
      GO TO 99
25    CONTINUE
      CALL FUN418       /* VIEW WITH SOLID, TRANSPARENT AND GS OBJECTS
      GO TO 99
26    CONTINUE
      CALL FUN410  /* 3D VIEW OF UNIVERSE
      GO TO 99
27    CONTINUE
      CALL FUN405  /* LOGICAL OPERATIONS
      GO TO 99
28    CONTINUE
      CALL FUN408  /* PRINT TABLES ON SCREEN
      GO TO 99
29    CONTINUE
      CALL FUN406  /* SAVE/RESTORE ITEM TO/FROM DISC
      GO TO 99
30    CONTINUE
      CALL FUN5                 /* CHANGE DEFAULT PARAMETERS
      GO TO 99
31    CONTINUE
C
C  NOTE: EXIT IS IMPLEMENTED IN SELECT AND SELCT2. THIS ALLOWS THE USER TO
C        EXIT WHENEVER THE LIGHTPEN IS ACTIVE. DO NOT PUT ANY CODE IN HERE
C        OR TRY TO CHANGE THE FUNCTION OF KEY 31.
C
      GO TO 99
C
      END
C
C
```

Appendix I - OCTREE Insert File (GP/1.INSERT)

```
C// GP/1.INSERT 203 GP/1 INSERT FILE FOR PROGRAM OCTREE
C
CI ------------------------------------------------------------------------
CI
CI     PROGRAM "GP/1" INSERT FILE
CI
CI IN THE SOURCE OF GP/1 PROGRAMS THIS FILE IS INSERTED WHEREVER THE
CI STATEMENT "$INSERT GP/1.INSERT" APPEARS BEGINNING IN COLUMN 1.
CI
CI ------------------------------------------------------------------------
CI
C    COPYRIGHT 1979, MEAGER SOFTWARE LIMITED, LOUDONVILLE, NEW YORK
CI
CI  BEGIN INSERT FILE
CI
CI
CI
      INTEGER POINT,LINE,CIRARC,CHAIN,AREA,FUNKEY,ERROR,ACK,MESSAG
      INTEGER MENU,FUNCTN,LEVEL,NLINES,GRID,TEMPIC
      INTEGER*4 IQUED
      INTEGER*4 DFSUB$,PROJ$
      INTEGER*2 ICHVTB,IARVTB
      LOGICAL NUMFRE
CI
CI
      DOUBLE PRECISION FUN(32)
CI
CI  PARAMETER STATEMENTS - DEFINE THE SIZE OF TABLES
CI
      PARAMETER (M$PIC=128)     /* MAXIMUM NUMBER OF SUBPICTURES
      PARAMETER (M$PNT=10)      /* MAXIMUM NUMBER OF POINTS
      PARAMETER (M$LIN=5)       /* MAXIMUM NUMBER OF LINES
      PARAMETER (M$CIR=2)       /* MAXIMUM NUMBER OF CIRCLES
      PARAMETER (M$CH=10)       /* MAXIMUM NUMBER OF CHAINS
      PARAMETER (M$CHV=1000)    /* MAXIMUM NUMBER OF CHAIN VALUES (LINKS)
      PARAMETER (M$AR=10)       /* MAXIMUM NUMBER OF AREA ENCLODED BLOBS
      PARAMETER (M$ARV=800000)  /* MAXIMUM NUMBER OF AREA VALUES (AREAS)
      PARAMETER (M$SCN=200000)  /* MAXIMUM SIZE OF SCAN TABLE
      PARAMETER (M$QUE=200000)  /* MAXIMUM SIZE OF QUEUE
      PARAMETER (M$QD=200000)   /* MAXIMUM SIZE OF DOUBLE PRECISION QUEUE
CI
CI  PARAMETER STATEMENTS - DEFINE COLUMNS OF OBJECT TABLE (IARTB)
CI
      PARAMETER (HEADR$=1)      /* HEADER
      PARAMETER (LEVEL$=2)      /* LEVEL
      PARAMETER (START$=3)      /* START LOCATION IN IARVTB
      PARAMETER (NOELE$=4)      /* NUMBER OF ELEMENTS
      PARAMETER (ORGNX$=5)      /* X ORIGIN
      PARAMETER (ORGNY$=6)      /* Y ORIGIN
      PARAMETER (ORGNZ$=7)      /* Z ORIGIN
      PARAMETER (SCALX$=8)      /* X SCALE
```

```
      PARAMETER (SCALY$=9)       /* Y SCALE
      PARAMETER (SCALZ$=10)      /* Z SCALE
      PARAMETER (USER1$=11)      /* USER LOCATION 1
      PARAMETER (USER2$=12)      /* USER LOCATION 2
      PARAMETER (USER3$=13)      /* USER LOCATION 3
      PARAMETER (USER4$=14)      /* USER LOCATION 4
CI
      DIMENSION ISUBTB(M$PIC,2)                    /* SUBPICTURE TABLE (TYPE,ENTRY)
      DIMENSION IPTABL(M$PNT),PTABLE(M$PNT,3)      /*POINT TABLES (HEADER,X,Y,Z)
      DIMENSION ILINTB(M$LIN),FLINTB(M$LIN,6)      /* LINE TABLES (HEADER,X1,Y1,
CI                                                    Z1,X2,Y2,Z2)
      DIMENSION ICIRTB(M$CIR),CIRTB(M$CIR,5)       /* CIRCLE TABLES (HEADER,X,Y,R,A,B)
      DIMENSION ICHTB(M$CH,6)                      /* CHAIN TABLE (HEADER,LEVEL,ISTART,
CI                                                    NOELEM,IGEGNX,IBEGNY)
      DIMENSION ICHVTB(M$CHV)                      /* CHAIN VALUE TABLE
      DIMENSION IARTB(M$AR,14)                     /* OBJECT TABLE (HEADER,LEVEL,
CI                                                    ISTART,NOELEM,IORGNX,IORGNY,
CI                                                    IORGNZ,ISCALX,ISCALY,ISCALZ,
CI                                                    USER1,USER2,USER3,USER4)
CI    NOTE: THE ORIGIN VALUES (IORGNX,IORGNY,IORGNZ) ASSUME A UNIVERSE EDGE
CI          OF 64K. TO BRING THE VALUES INTO A 0 TO 1024 UNIVERSE, DIVIDE BY
CI          64 SHIFT RIGHT BY 6). THE SCALE VALUES REPRESENT THE NUMBER OF
CI          UNITS WHICH ARE REPRESENTED BY THE ASSOCIATED EDGE OF THE UNIVERSE.
CI          AN UNSCALED UNIVERSE HAS AN EDGE OF 64K. THE SCALE 1 VALUE IS THUS
CI          64K. A VALUE OF 0 IS ASSUMED TO BE A SCALE FACTOR OF 1 (64K IS
CI          USED). A VALUE GREATER THAN 64K MEANS THAT THE OBJECT IS LARGER
CI          IN THAT DIMENSION (IT WILL BE LARGER WHEN DISPLAYED). A SMALLER
CI          VALUE MEANS THAT AN OBJECT IS ACTUALLY SMALLER IN THAT DIMENSION.
CI          TO CONVERT TO A 1K TY 1K UNIVERSE, DIVIDE EACH VALUE BY 64 (SHIFT
CI          RIGHT BY 6 BIT POSITIONS).
CI                    NOTE: IN IARTB HEADER, BITS 0 - 2 ARE RESERVED FOR THE
CI                    NUMBER OF DIMENSIONS β BIT 3 IS A 0 FOR UNTAGGED AND A
CI                    1 FOR TAGGED. (BIT 0 IS THE LSB.)
CI                    CURRENT OBJECT HEADER BIT DEFINITIONS:
CI    BIT  MASK     MEANING
CI    0-2  :7       DIMENSIONS OVER WHICH OBJECT DEFINED (1 TO 7)
CI     3   :10      TAG BIT (0=NOT TAGGED, 1=TAGGED)
CI     4   :20      TAG TYPE (0=VECTOR, 1=RASTER)
CI     5   :40      TEST OBJECT (0=NOT TEST OBJECT, 1=TEST OBJECT)
CI     6   :100     DISPLAY INHIBIT (0=DISPLAY, 1=INHIBIT DISPLAY)
CI     7   :200     INTERFERENCE OBJECT (0=NO INTERFERENCE CHECK, 1=CHECK)
CI     8   :400     TRANSPARENT OBJECT (0=SOLID OBJECT, 1=TRANSPARENT OBJECT)
CI     9   :1000    ADDATIVE/SUBTRACTIVE OBJECT (0=ADDATIVE, 1=SUBTRACTIVE)
CI    10   :2000    GRAY SCALE OBJECT (0=HOMOGENEOUS OBJECT, 1=GRAY SCALE OBJECT)
CI    11   :4000    SURFACE NORMAL OBJECT (0=NOT SN, 1=SN OBJECT)
CI
      DIMENSION IARVTB(M$ARV)              /* AREA VALUE TABLE
      DIMENSION ISCAN(M$SCN,2)             /* SCAN TABLE
      DIMENSION IQUEUE(M$QUE,2)            /* QUEUE
      DIMENSION IQUED(M$QD,2)              /* DOUBLE PRECISION QUEUE (INTEGER*4)
CI
CI
      DIMENSION ICHAR$(2)                  /* TEMPORARY STORAGE FOR EIGHT A/N
CI                                            CHARACTERS (FOR ENCODE, ETC.)
```

```
      DIMENSION DFSUB$(8)                  /* NAME OF DATA FILE SUB-UFD
      DIMENSION DIMAG$(8)                  /* NAME OF IMAGE FILE SUB-UFD
      DIMENSION DGRPH$(8)                  /* NAME OF GRAPHICS FILE SUB-UFD
      DIMENSION PROJ$(8)                   /* NAME OF PROJECT (USED TO GENERATE NAMES
CI                                            NAMES OF DATA FILES)
      DIMENSION IBUF$(25)                    /* TEMPORARY STORAGE FOR UP TO 100
CI                                           CHARACTERS (FOR READ, WRITE, ETC.)
CI
      COMMON/AREA/ISUBTB,IPTABL,PTABLE,ILINTB,FLINTB,ICIRTB,CIRTB,
     2 MAXPIC,POINT,LINE,CIRARC,IDMARK,FUNCTN,ERROR,ACK,MESSAG,MENU,
     3 FUNKEY,FUN,CHAIN,AREA,LEVEL,USCALE,XORIGN,YORIGN,NLINES,
     4 ICHTB,IARTB,MAXCH,MAXCHV,MAXAR,MAXARV,
     5 MAXSCN,LEVSCN,NPOINT,NHASH,HDELTA,IQR,IQF,MAXQUE,IGRID,
     6 GRID,NDELAY,INTENS,LCHINS,TEMPIC,IQFD,IQRD,MAXQD,PI,ITABLT,
     7 COL$PW,COL$OF,DFSUB$,PROJ$,I$PART,IBACK,MAXCIR,MAXLIN,MAXPNT,
     8 GSMLT$,SLMLT$,IBCK$0,IBCK$1,IBCK$2,IBUF$,DIMAG$,DGRPH$
CI
      COMMON/ARV/IARVTB
CI
      COMMON/CHV/ICHVTB
CI
      COMMON/QUE/IQUEUE
      COMMON/QUE2/IQUED
      COMMON/SCANCM/ISCAN
CI
CI
      DATA MAXPIC/M$PIC/     /* MAXIMUM NUMBER OF SUBPICTURES
      DATA MAXPNT/M$PNT/     /* MAXIMUM NUMBER OF POINTS
      DATA MAXLIN/M$LIN/     /* MAXIMUM NUMBER OF LINES
      DATA MAXCIR/M$CIR/     /* MAXIMUM NUMBER OF CIRCLES
      DATA MAXCH/M$CH/       /* MAXIMUM NUMBER OF CHAINS
      DATA MAXCHV/M$CHV/     /* MAXIMUM NUMBER OF CHAIN VALUES (LINKS)
      DATA MAXAR/M$AR/       /* MAXIMUM NUMBER OF AREA ENCLODED BLOBS
      DATA MAXARV /M$ARV/    /* MAXIMUM NUMBER OF AREA ENCODED VALUES (AREAS)
      DATA MAXSCN/M$SCN/     /* MAXIMUM SIZE OF SCAN TABLE
      DATA MAXQUE/M$QUE/     /* MAXIMUM SIZE OF QUEUE
      DATA MAXQD /M$QD/      /* MAXIMUM SIZE OF DOUBLE PRECISION QUEUE
CI
CI  DEFINITION OF SUBPICTURE TYPES
CI
      DATA POINT/1/          /* POINT
      DATA LINE/2/           /* LINE
      DATA CIRARC/3/         /* CIRCLE/ARC
      DATA CHAIN/4/          /* CHAIN ENCODED BLOB
      DATA AREA/5/           /* AREA ENCODED BLOB
      DATA FUNKEY/100/       /* FUNCTION KEY 0
      DATA IBACK/150/        /* BACKGROUND SUBPICTURE
      DATA ERROR/155/        /* ERROR MESSAGE
      DATA ACK/156/          /* ACKNOWLEDGEMENT MESSAGE
      DATA MESSAG/170/       /* MESSAGE (FIRST)
      DATA MENU/185/         /* MENU (FIRST)
      DATA FUNCTN/151/       /* FUNCTION MESSAGE
      DATA GRID/161/         /* GRID LINES
      DATA TEMPIC/250/       /* TEMPORARY SUBPICTURE (CAN BE DELETED OR ERASED)
```

```
CI
CI  DEFAULTS
      DATA IDMARK/4/           /* DEFAULT POINT MARKER (4=CROSS)
      DATA LEVEL/5/            /* DEFAULT LEVEL
      DATA NLINES/32/          /* DEFAULT 2**LEVEL VALUE
      DATA XORIGN,YORIGN/-4.5,-4.5/  /* DEFAULT ORIGIN
      DATA USCALE/1.75781/     /* DEFAULT UNIVERSAL SCALE (9 IN VIEWABLE/(1024/200))
      DATA LEVSCN/1/           /* LEVEL REPRESENTED BY THE VALUES CURRENTLY IN
CI                                THE SCAN TABLE ISCAN
      DATA NPOINT/0/           /* NUMBER OF POINTS IN THE SCAN TABLE
      DATA NHASH/3/            /* LEVEL OF GRID LINES (POINTS BETWEEN = 2**NHASH)
      DATA IQF/0/              /* FRONT QUEUE POINTER
      DATA IQR/0/              /* REAR QUEUE POINTER
      DATA IQFD/0/             /* DOUBLE PRECISION QUEUE - FRONT POINTER
      DATA IQRD/0/             /* DOUBLE PRECISION QUEUE - REAR POINTER
      DATA NDELAY  /0/         /* DELAY(IN MSEC) WHEN PUTTING OUT
CI                                AN AREA SQUARE IN SUBROUTINE DSPAR
      DATA INTENS/4/           /* DEFAULT INTENSITY FOR AREA DISPLAY
      DATA LCHINS/10/          /* DEFAULT INTENSITY FOR CHAIN DISPLAY
      DATA ITABLT/0/           /* DEFAULT TABLET (BITPAD) STATUS (0 IS UNITIALIZED)
      DATA I$PART/0/           /* DEFAULT VALUE FOR PARTIAL SQUARE DISPLAY
      DATA PI/3.14159/         /* THE VALUE OF PI
      DATA COL$PW/1.0/         /* DEFAULT EXPONENT FOR COLOR VALUE (SEE INITT)
      DATA COL$OF/0.0/         /* DEFAULT OFFSET FOR COLOR VALUE
      DATA DFSUB$/'S_OCTREE_DATA_FILES              '/ /* NAME OF DATA SUB-UFD
      DATA DIMAG$/'S_OCTREE_IMAGE_FILES             '/ /* NAME OF IMAGE SUB-UFD
      DATA DGRPH$/'S_OCTREE_GRAPHICS_FILES          '/ /* NAME OF GRAPHICS SUB-UFD
      DATA PROJ$ /'D_MISC                           '/ /* PROJECT NAME FOR DATA FILES
      DATA GSMLT$/0.1/                  /* GRAY SCALE OBJECT INTENSITY MULTIPLIER
GS)
      DATA SLMLT$/100/           /* SOLID OBJECT INTENSITY MULTIPLIER (SEE DZ3DGS)
      DATA IBCK$0/0/                /* SUBPIXEL ARRAY INITIALIZATION - CHAN 0
      DATA IBCK$1/0/                /* SUBPIXEL ARRAY INITIALIZATION - CHAN 1
      DATA IBCK$2/0/                /* SUBPIXEL ARRAY INITIALIZATION - CHAN 2
CI
C********************************************************* END OF BLOCK *******
C
```

Appendix J - Program OCTREE Subroutines

ROW 224 DETERMINE ROW STATUS

```
  SUBROUTINE ROW(IX1,IX2,IRSTAT,IROWPT)
  THIS SUBROUTINE RETURNS THE STATUS OF A SECTION OF A SPECIFIED ROW
  BETWEEN IX1 AND IX2 RELATIVE TO THE ISCAN TABLE. IROWPT POINTS TO THE
  THE ENTRY IN ISCAN WHICH IS FOR THE ROW IN QUESTION (Y VALUE IS AT
  ISCAN TABLE LEVEL, NOT NECESSARILY AT LEVEL 0). IX1 AND IX2 ARE THE
  TWO ENDS (IX1.LE.IX2). IX1 IS THE X VALUE OF THE ORIGIN OF THE FIRST
  SQUARE IN QUESTION, WHILE IX2 IS THE ORIGIN OF THE LAST. NOTE
  THAT THE RIGHT END OF THE SECTION IS IX2+1, NOT IX2. THE STATUS OF
  THE ROW IS RETURNED IN IRSTAT (0=EMPTY, 1=PARTIAL, 2=FULL).
```

LOOK 225 LOOK AT AN AREA

```
  SUBROUTINE LOOK(IORIGX,IORIGY,ILEVEL,ISTAT)
  THIS SUBROUTINE EXAMINES THE SCAN TABLE ISCAN TO DETERMINE IF THE
  INDICATED SQUARE AT LEVEL ILEVEL AND ORIGIN IORIGX, IORIGY (ABSOLUTE,
  LEVEL 0 VALUES) IS EMPTY, PARTIALLY FULL OR FULL.
  ISTAT IS RETURNED AS 0 FOR EMPTY, 1 FOR PARTIAL AND 2 FOR FULL.
```

SQUARE 226 DRAW SQUARE

```
  SUBROUTINE SQUARE(IX,IY,ILEVEL,NTYPES)
  THIS SUBROUTINE DRAWS A SQUARE INTO THE OPEN SUBPICTURE AT IX,IY (LEVEL
  0 VALUES). IT IS DRAWN AT A SCALE CORRESPONDING TO ILEVEL. IF NTYPES=0
  NOTHING IS DRAWN. IF NTYPES=1, A SIMPLE SQUARE IS DRAWN. IF NTYPES=2, THE
  SQUARE IS FILLED IN WITH LINES AT 45 DEGREES. THE DIFFERENCE IN X AND Y
  BETWEEN THE 45 DEGREE LINES IS 2**NHASH (IN COMMON) LINES (LEVEL 0).
  ADDITION: IF NTYPES .GT. 2 THE LINES ARE AT 135 DEGREES
```

CIRCH 227 CIRCLE CHAIN INPUT

```
  SUBROUTINE  CIRCH(ICALL,ICH,IHEADR,CENTX,CENTY,RADIUS)
  THE PARAMETERS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      ICH - CHAIN NUMBER (RETURNED TO CALLING PROGRAM)
      IHEADR - HEADER FOR NEW CHAIN TO GENERATE A CIRCULAR CHAIN AND PUT IT
               INTO THE TABLES.
      CENTX - X VALUE OF CIRCLE (-4.5 TO 4.5)
      CENTY - Y VALUE OF CENTER
      RADIUS - RADIUS OF CIRCLE
```

SLICE 228 CREATE A SLICE FROM AN AREA BLOB

```
  SUBROUTINE SLICE(NOAREA,IARNUM,IHEADR,ISLICE)
  THIS SUBROUTINE WILL GENERATE A NEW VOLUME ENCODED SLICE FROM AN AREA (2D)
```

BLOB. THE SLICE WILL HAVE A DEPTH EQUAL TO AN EDGE SIZE OF A SQUARE AT THE CURRENT LEVEL. THE EXACT SLICE IS PASSED IN ISLICE (BEGINNING WITH 0).

NOAREA - THE IARTB POINTER OF THE ITEM
IARNUM - THE NEW AREA IARTB POINTER (RETURNED)
IHEADR - HEADER OF THE NEW AREA
ISLICE - THE NUMBER OF THE SLICE (0 TO N)

INSARV 228.1 ADD AN ITEM TO THE AREA VALUE TABLE

SUBROUTINE INSARV(IVAL,IARVPT,IARNUM)
THIS SUBROUTINE IS SIMILIAR TO ADDARV EXCEPT IT MARKS -1 AT THE END AND INCREMENTS THE COUNT EACH TIME.

TOAREA 229 CONVERT TO AREA (FROM CHAIN CODE)

SUBROUTINE TOAREA(ICALL,IHEADR,ICHNUM,IARNUM,NLEVEL)
SPECIFY: ICALL - CALL NUMBER
ICHNUM - CHAIN NUMBER TO CONVERT (POINTER TO ICHTB)
IARNUM - AREA CODE NUMBER (POINTER IN IARTB)
IHEADR - HEADER VALUE FOR THE AREA CODE
NLEVEL - MINIMUM LEVEL TO GENERATE AREA CODE
IF AN AREA CODE EQUAL TO IARNUM ALREADY EXISTS, IT WILL BE DELETED (ALTHOUGH THE AREA VALUES IN IARVTB WILL REMAIN). IF NLEVEL IS LESS THAN THE CURRENT SCAN TABLE LEVEL, IT WILL BE DIGITIZED TO THIS LEVEL (PARTIALS WILL BE FULL AT THE LOWEST LEVEL).
NOTE: THE SQUARES ARE NOT PUT INTO A SUBPICTURE.
THE SQUARES ARE ARRANGED AS FOLLOWS:

3 4
1 2

DSPAR 230 DISPLAY AREA

SUBROUTINE DSPAR(NOAREA,LEVREQ)
THIS SUBROUTINE WILL DISPLAY AN AREA ENCODED BLOB AS A SERIES OF SQUARES. THE NUMBER OF THE AREA IS SPECIFIED IN NOAREA (NUMBER WHICH POINTS TO ENTRY IN IARTB). LEVREQ IS THE REQUESTED LEVEL AT WHICH THE BOXES SHOULD BE DISPLAYED. IF LEVREQ = -1, ALL LEVELS FROM 10 DOWN TO LEVREQ WILL BE DISPLAYED. AT ANY LEVEL, NOTHING IS DISPLAYED FOR AN EMPTY BOX, AN EMPTY SQUARE IS WRITTEN FOR A PARTIAL SQUARE AND A FILLED IN SQUARE (SEE SUBROUTINE 'SQUARE') IS DISPLAYED FOR A FULL SQUARE.
A SUBPICTURE IS DEFINED FOR THE PRESENTATION. IT POINTS TO THE ENTRY IN THE AREA TABLE.

DSPCH 231 DISPLAY CHAIN

SUBROUTINE DSPCH(NCHAIN)
THIS SUBROUTINE WILL DISPLAY A CHAIN. NCHAIN IS THE NUMBER OF THE CHAIN (POINTER TO ENTRY IN ICHTB). A SUBPICTURE IS DEFINED WHICH POINTS TO THE ENTRY IN THE AREA TABLE.

MATCH 232 FIND MATCH BETWEEN 2 AREAS

SUBROUTINE MATCH(IARNO1,IARNO2,NLEVEL)
THIS SUBROUTINE WORKS THROUGH TWO AREA ENCODED BLOBS AND DISPLAYS
THE AREAS OF COMPLETE INTERSECTION WITH FILLED IN SQUARES (HASH MARKS
AT 135 DEGREES). IARNO1 AND IARNO2 ARE THE NUMBERS OF THE TWO AREAS
(POINTERS TO THE IARTB TABLE).

AMOVE 233 MOVE (TRANSLATE) AREA

SUBROUTINE AMOVE(ICALL,NOAREA,MOVEX,MOVEY,IHEADR,IARNUM,MINLEV)
THIS SUBROUTINE WILL CONVERT AN AREA ENCODED BLOB INTO A SECOND WHICH IS
A TRANSLATED VERSION OF THE FIRST. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
ICALL - CALL NUMBER
NOAREA - THE IARTB POINTER OF THE AREA TO BE TRANSLATED
MOVEX - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE X
DIRECTION (POSITIVE NUMBER ONLY)
MOVEY - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE Y
DIRECTION (POSITIVE NUMBER ONLY)
IARNUM - THE IARTB POINTER FOR THE NEW (TRANSLATED) AREA
IHEADR - THE HEADER VALUE FOR THE NEW (TRANSLATED) AREA
MINLEV - MINIMUM LEVEL OF NEW AREA (BECAUSE THE NEW UNIVERSE PROBABLY
WILL NOT COMPLETELY ALLIGN WITH THE OLD UNIVERSE, THE PROCESS
MAY NEVER CONVERGE FOR SOME SQUARES). TYPICALLY SET TO LEVEL
OF OLD UNIVERSE OR ONE OR TWO BELOW.
NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

AROT90 234 ROTATION OF AN AREA BY 90 DEGREES

SUBROUTINE AROT90(ICALL,NOAREA,ISEL,IARNUM,IHEADR)
THIS SUBROUTINE USES THE 4 VALUED ARRAY ISEL TO REARRANGE THE ORDER OF AN
AREA ENCODED BLOB. THIS CAN BE USED TO CAUSE 90 DEGREE ROTATION OR
REFLECTION OF AN OBJECT. THE TRANSFORMED AREA IS PUT INTO A SECOND AREA
AS SPECIFIED BY THE CALLING PROGRAM.
THE ARGUMENTS ARE DEFINED AS FOLLOWS:
ICALL - CALL NUMBER
NOAREA - THE IARTB POINTER OF THE AREA TO BE TRANSLATED
ISEL - (DIMENSION ISEL(4)) GIVES THE ORDER FOR THE NEW AREA. THE
NUMBER OF THE NEW SON 0 IS SPECIFIED IN ISEL(1), AND SO ON.
IARNUM - THE IARTB POINTER FOR THE NEW (TRANSLATED) AREA
IHEADR - THE HEADER VALUE FOR THE NEW (TRANSLATED) AREA
NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

FUN400 235 90 DEGREE ROTATION

SUBROUTINE FUN400

AROT 236 ROTATE AN AREA - (0 TO PI/2)

```
  SUBROUTINE AROT(ICALL,NOAREA,MOVEX,MOVEY,ANG,IARNUM,IHEADR,MINLEV)
  THIS SUBROUTINE WILL CONVERT AN AREA ENCODED BLOB INTO A SECOND WHICH IS
  A ROTATED VERSION OF THE FIRST. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      NOAREA - THE IARTB POINTER OF THE AREA TO BE ROTATED
      MOVEX - THE LEVEL 0 VALUE AT WHICH THE AREA IS TO BE ROTATED (IN THE X
              DIRECTION) (POSITIVE NUMBER ONLY)
      MOVEY - THE LEVEL 0 VALUE AT WHICH THE AREA IS TO BE ROTATED (IN THE Y
              DIRECTION) (POSITIVE NUMBER ONLY)
      ANG - ROTATION ANGLE (RADIANS) OF THE NEW UNIVERSE (OBJECTS MOVED BY
              -ANG IN THE NEW UNIVERSE). THIS IS A FLOATING POINT VALUE
              BETWEEN 0 AND PI/2.
      IARNUM - THE IARTB POINTER FOR THE NEW (ROTATED) AREA
      IHEADR - THE HEADER VALUE FOR THE NEW (ROTATED) AREA
      MINLEV - MINIMUM LEVEL OF NEW AREA (BECAUSE THE NEW UNIVERSE PROBABLY
               WILL NOT COMPLETELY ALLIGN WITH THE OLD UNIVERSE, THE PROCESS
               MAY NEVER CONVERGE FOR SOME SQUARES). TYPICALLY SET TO LEVEL
               OF OLD UNIVERSE OR ONE OR TWO BELOW.
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

FUN401 237 0 TO 90 DEGREE ROTATION

  SUBROUTINE FUN401

FUN411 238 FUNCTION TO CREATE A SLICE FROM AN AREA

  SUBROUTINE FUN411

VIEW3P 239 3-D VECTOR DISPLAY WITH PERSPECTIVE

  SUBROUTINE VIEW3P(ICALL,IOPTU,IFNCT2,IARARY,IAREAS,IPLANE,
  ANG1,ANG2,ANG3,SCALE,IDIST2,IMOVEX,IMOVEY,NSUBUN,NSUBVW)
  THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP
  TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
       ICALL - CALL NUMBER
       IFNCT2 - DESIRED FUNCTION
                  1 - VIEW FROM INFINITE DISTANCE
                  2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
       IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                TO BE DISPLAYED (IARTB POINTERS)
       IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
       IPLANE - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE
                (1=X,2=Y,3=Z)
       ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE
             TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2
             OF FROM 0 TO 90 DEGREES).
       IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE
               VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
       IMOVEX - MOVE THE CENTER OF THE VIEWPLANE BY THIS DITANCE (LEVEL
                0 UNITS IN THE X DIRECTION (SCREEN COORDINATES)
```

IMOVEY - SAME FOR Y
NSUBUN - SUBPICTURE NUMBER OF UNIVERSE (IF ANY) (RETURNED)
NSUBVW - SUBPICTURE NUMBER OF VIEW (RETURNED)
NOTE: IF ONLY ONE OBJECT DISPLAYED, IT HAS A TYPE OF 'AREA' AND IS IDENTIFIED IN THE SUBPICTURE TABLE (CAN BE DELETED LATER).

EDGES ARE DISPLAYED FROM THE ORIGIN OUT (AS WOULD BE VIEWED FROM A POINT FROM 0 TO 90 DEGREES.
POINTS ARE NUMBERED AS FOLLOWS:

```
      Y
      .
      .
      3   4
                              AT Z=0 (Z INTO THE PAGE, LH COORD SYS)
      1   2 .. X
      7   8
                              AT Z=1
      5   6
```

EDGE 240 DISPLAY THE EDGES OF AN AREA

SUBROUTINE EDGE(ICALL,NOAREA)
THIS SUBROUTINE WILL DISPLAY THE EDGES (EXTERNAL AND INTERNAL) OF AN AREA ENCODED ITEM. THE ARGUMENTS ARE AS FOLLOWS:
ICALL - CALL NUMBER
NOAREA - THE IARTB POINTER OF THE AREA TO BE DISPLAYED
THE EDGES TO BE DISPLAYED ARE WRITTEN INTO A TEMPORARY SUBPICTURE (ITYPE=TEMPIC). NO DATA TABLE IS GENERATED.
NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES.

FUN402 241 DISPLAY EDGES OR GENERATE SHELL

SUBROUTINE FUN402

VIEW2 242 DISPLAY THE SQUARES OF AN AREA IN SORTED ORDER

SUBROUTINE VIEW2(ICALL,NOAREA)
THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF AN AREA ENCODED ITEM IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
ICALL - CALL NUMBER
NOAREA - THE IARTB POINTER OF THE AREA TO BE DISPLAYED
EDGES ARE DISPLAYED FROM THE ORIGIN OUT (AS WOULD BE VIEWED FROM A POINT FROM 0 TO 90 DEGREES.
THE EDGES TO BE DISPLAYED ARE WRITTEN INTO A TEMPORARY SUBPICTURE (ITYPE=TEMPIC). NO DATA TABLE IS GENERATED.

FUN403 243 DISPLAY THE EDGES OF AN AREA IN ORDER

SUBROUTINE FUN403

MVIEW2 244 DISPLAY THE SQUARES OF MULTIPLE AREAS IN SORTED ORDER

```
  SUBROUTINE MVIEW2(ICALL,IARARY,IAREAS)
  THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP
  TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
       ICALL - CALL NUMBER
       IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                TO BE DISPLAYED (IARTB POINTERS)
       IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
  EDGES ARE DISPLAYED FROM THE ORIGIN OUT (AS WOULD BE VIEWED FROM A
  POINT FROM 0 TO 90 DEGREES.
  THE EDGES TO BE DISPLAYED ARE WRITTEN INTO A TEMPORARY SUBPICTURE (ITYPE=
  TEMPIC). NO DATA TABLE IS GENERATED.
```

SHELL 245 CREATE A VISIBLE SHELL AREA

```
  SUBROUTINE SHELL(ICALL,NOAREA,IARNUM,IHEADR)
  THIS SUBROUTINE WILL GENERATE A NEW AREA ENCODED BLOB WITH ONLY THE
  POSSIBLY VISIBLE SQUARES OF AN AREA. AT THIS TIME, THIS MEANS VISIBLE TO
  AN OBSERVER LOCATED SOMEWHERE BETWEEN 0 AND 90 DEGREES.
  THE ARGUMENTS ARE AS FOLLOWS:
       ICALL - CALL NUMBER
       NOAREA - THE IARTB POINTER OF THE AREA TO BE DISPLAYED
       IARNUM - THE NEW AREA IARTB POINTER
       IHEADR - HEADER OF THE NEW AREA
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES.
```

VIEW2P 246 2-D DISPLAY WITH PERSPECTIVE

```
  SUBROUTINE VIEW2P(ICALL,IFUNCT,IARARY,IAREAS,IPLANX,IPLANY,ANG,
      IDIST)
  THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP
  TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
       ICALL - CALL NUMBER
       IFUNCT - DESIRED FUNCTION
                  1 - VIEW FROM INFINITE DISTANCE
                  2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
       IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                TO BE DISPLAYED (IARTB POINTERS)
       IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
       IPLANX - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN X
       IPLANY - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN Y
       ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE
             TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2 OF
             FROM 0 TO 90 DEGREES).
       IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE
               VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
  EDGES ARE DISPLAYED FROM THE ORIGIN OUT (AS WOULD BE VIEWED FROM A
  POINT FROM 0 TO 90 DEGREES.
```

```
EXPAND 247 EXPAND AN N-DIMENSIONAL OBJECT INTO N+1 DIMENSIONS

  SUBROUTINE EXPAND(ICALL,NOAREA,IDIMEN,IARNUM,IHEADR)
  THIS SUBROUTINE WILL EXPAND AN N-DIIMENSIONAL OBJECT INTO AN N+1
  DIMENSIONAL OBJECT. THUS, FOR EXAMPLE, A 2D CIRCLE BECOMES A CYLINDER OR
  A 2D POLYGON BECOMES A PRISM. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
    ICALL - CALL NUMBER
    NOAREA - THE IARTB POINTER OF THE OBJECT
    IDIMEN - THE DIMENSION OF THE OBJECT REPRESENTED BY NOAREA (A SQUARE OR
             CIRCLE IS 2, A CUBE IS 3, ETC.)
    IARNUM - THE IARTB POINTER FOR THE NEW OBJECT
    IHEADR - THE HEADER VALUE FOR THE NEW OBJECT

FUN404 248 EXPAND AN ITEM

  SUBROUTINE FUN404

VIEW3D 249 DISPLAY THE SQUARES OF MULTIPLE AREAS IN SORTED ORDER

  SUBROUTINE VIEW3D(ICALL,IOPTU,IFUNCT,IARARY,IAREAS,IPLANX,IPLANY,
      IPLANZ,ANG1,ANG2,ANG3,SCALE,IDIST,NSUBUN,NSUBVW)
  THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP
  TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
       ICALL - CALL NUMBER
       IFUNCT - DESIRED FUNCTION
                  1 - VIEW FROM INFINITE DISTANCE
                  2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
       IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                TO BE DISPLAYED (IARTB POINTERS)
       IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
       IPLANX - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN X
       IPLANY - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN Y
       ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE
             TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2
             OF FROM 0 TO 90 DEGREES).
       IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE
               VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
       NSUBUN - SUBPICTURE NUMBER OF UNIVERSE (IF ANY) (RETURNED)
       NSUBVW - SUBPICTURE NUMBER OF VIEW (RETURNED)
                NOTE: IF ONLY ONE OBJECT IS DISPLAYED, IT HAS A TYPE OF AREA
                IN THE SUBPICTURE TABLE AND THE OBJECT CAN LATER BE DELETED.
  EDGES ARE DISPLAYED FROM THE ORIGIN OUT (AS WOULD BE VIEWED FROM A
  POINT FROM 0 TO 90 DEGREES.

FUN410 250 OBJECT DISPLAY FUNTION KEY

  SUBROUTINE FUN410

GETANG 250.1 OBTAIN VIEW ANGLES FROM USER
```

```
  SUBROUTINE GETANG(ANG1,ANG2,ANG3)

VROT90 251 ROTATION OF A VOLUME BY 90 DEGREES

  SUBROUTINE VROT90(I ALL,NOAREA,ISEL,IARNUM,IHEADR)
  THIS SUBROUTINE USES THE 8 VALUED ARRAY ISEL TO REARRANGE THE ORDER OF AN
  OBJECT. THIS CAN BE USED TO CAUSE 90 DEGREE ROTATION OR
  REFLECTION OF AN OBJECT. THE TRANSFORMED OBJECT IS PUT INTO A SECOND
  OBJECT AS SPECIFIED BY THE CALLING PROGRAM.
  NOTE: THE ASSOCIATED IARTB INFORMATION (5 AND UP) IS SIMPLY
        COPIED TO THE NEW OBJECT.
  THE ARGUMENTS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      NOAREA - THE IARTB POINTER OF THE AREA TO BE TRANSLATED
               (MUST BE SPECIFIED BY CALLING PROGRAM)
      ISEL - (DIMENSION ISEL(8)) GIVES THE ORDER FOR THE NEW AREA. THE
             NUMBER OF THE NEW SON 0 IS SPECIFIED IN ISEL(1), AND SO ON.
      IARNUM - THE IARTB POINTER FOR THE NEW (TRANSLATED) AREA
               (MUST BE SPECIFIED)
      IHEADR - THE HEADER VALUE FOR THE NEW (TRANSLATED) AREA
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

INTERS 252 GENERATE THE INTERSECTION OF TWO ITEMS

  SUBROUTINE INTERS(IDIMEN,NOARA,NOARB,IARNUM,IHEADR)
  THIS SUBROUTINE WILL ACCEPT TWO ITEMS, A AND B AND WILL GENERATE A THIRD
  ITEM, C, WHICH IS THE INTERSECTION OF A AND B. THE ARGUMENTS ARE DEFINED
  AS FOLLOWS:
    IDIMEN - THE DIMENSION OF THE OBJECTS (A 2D OBJECT IS 2, 3D IS 3, ETC.)
    NOARA  - THE IARTB POINTER FOR ITEM A
    NOARB  - THE IARTB POINTER FOR ITEM B
    IARNUM - THE IARTB POINTER FOR ITEM C (TO BE GENERATED) (SPECIFIED BY
             CALLING PROGRAM).
    IHEADR - THE HEADER VALUE FOR THE NEW OBJECT

UNION 252.1 GENERATE THE UNION OF TWO ITEMS

  SUBROUTINE UNION(IDIMEN,NOARA,NOARB,IARNUM,IHEADR)
  THIS SUBROUTINE WILL ACCEPT TWO ITEMS, A AND B AND WILL GENERATE A THIRD
  ITEM, C, WHICH IS THE UNION OF A AND B. THE ARGUMENTS ARE DEFINED
  AS FOLLOWS:
    IDIMEN - THE DIMENSION OF THE OBJECTS (A 2D OBJECT IS 2, 3D IS 3, ETC.)
    NOARA  - THE IARTB POINTER FOR ITEM A
    NOARB  - THE IARTB POINTER FOR ITEM B
    IARNUM - THE IARTB POINTER FOR ITEM C (TO BE GENERATED) (SPECIFIED BY
             CALLING PROGRAM)
    IHEADR - THE HEADER VALUE FOR THE NEW OBJECT

SUBTRA 252.2 GENERATE THE SUBTRACTION OF TWO ITEMS
```

```
  SUBROUTINE SUBTRA(IDIMEN,NOARA,NOARB,IARNUM,IHEADR)
  THIS SUBROUTINE WILL ACCEPT TWO ITEMS, A AND B AND WILL GENERATE A THIRD
  ITEM, C, WHICH IS THE SUBTRACTION OF A AND B. THE ARGUMENTS ARE DEFINED
  AS FOLLOWS:
    IDIMEN - THE DIMENSION OF THE OBJECTS (A 2D OBJECT IS 2, 3D IS 3, ETC.)
    NOARA  - THE IARTB POINTER FOR ITEM A
    NOARB  - THE IARTB POINTER FOR ITEM B
    IARNUM - THE IARTB POINTER FOR ITEM C (TO BE GENERATED) (SPECIFIED BY
             CALLING PROGRAM)
    IHEADR - THE HEADER VALUE FOR THE NEW OBJECT

QPUT 252.3 BUFFER AND PUT VALUES INTO THE QUEUE

  SUBROUTINE QPUT(IVAL)
  WHEN A VALUE IS REQUESTED TO BE INSERTED IN THE QUEUE, A CHECK IS MADE TO
  SEE IF THE BUFFER HAS AN ENTRY (IT'S NOT 0). IF IT'S THE SAME, THE NUMBER
  IS SIMPLY INCREMENTED. IF DIFFERENT, THE OLD VALUE IS INSERTED
  AND THE NEW VALUE IS PLACED IN THE BUFFER. IF A VALUE OF 0 IS REQUESTED,
  THE BUFFER IS FLUSHED (IF NOT 0).

ADDARV 252.4 ADD AN AREA VALUE TO AN ITEM AT IARVPT

  SUBROUTINE ADDARV(IVAL,IARVPT)

FUN405 253 LOGICAL OPERATIONS FUNCTION BUTTON

  SUBROUTINE FUN405

PARAL 254 DRAW A PARALLELOGRAM ON THE DEANZA DISPLAY

  SUBROUTINE PARAL(ITEM,ISIDE,IVERTX)
  THIS SUBROUTINE WILL PUT A PARALLELOGRAM ON THE DEANZA DISPLAY. THE
  ARGUMENTS ARE DEFINED AS FOLLOWS:
    ITEM - THE NUMBER OF THE OBJECT BEING DISPLAYED (1 TO 10)
    ISIDE - THE SIDE BEING DISPLAYED (1 TO 3)
    IVERTX - AN ARRAY (4,2) SPECIFYING THE X,Y LOCATIONS OF THE FOUR
             CORNERS OF THE PARALLELOGRAM IN CLOCKWISE ORDER BEGINNING
             WITH THE POINT WITH THE LOWEST Y VALUE (0<VALUE<511)

ITTL 254.1 LOAD THE COLOR TABLE

  SUBROUTINE ITTL(ITEM,ISIDE,ICHAN,IVAL)

INIT 254.2 INITIALIZE DEANZA

  SUBROUTINE INIT(IOPT)
  IF IOPT=1, ENABLE ITT'S. DISABLE OTHERWISE.

MATT 254.3 FILLS ALL FOUR CHANNELS WITH CONSTANT
```

```
  SUBROUTINE MATT(I0,I1,I2,I3)
                                        I0 = INTENSITY CHANNEL 0
                                        I1 = INTENSITY CHANNEL 1
                                           ETC.
                          INTENSITY LEVELS ARE FROM 0 TO 255

LINE 254.4 PUTS ONE 512 BYTE LINE ON SCREEN

  SUBROUTINE LINE(CHAN,VERT,BUFF)
                          CHAN IS IMAGE CHANNEL NUMBER (0-3)
                          VERT IS VERTICAL LINE NUMBER (0-511)
                              (0 IS BOTTOM LINE)
                          BUFF IS A 256 WORD BUFFER WITH PIXEL INFO

DZLINE 254.5 PUTS A LINE SEGMENT ON SCREEN

  SUBROUTINE DZLINE(CHAN,IX,IY,LOCBUF,LENVAL)
                          CHAN IS IMAGE CHANNEL NUMBER (0-3)
                          IX IS X STARTING POSITION (0-511)
                          IY IS LINE NUMBER  (0-511)
                              (0 IS BOTTOM LINE)
                          LOCBUF IS THE LOC OF  UP TO 256 WORD BUFFER WITH
                          PIXEL INFO.

INITT 254.6 INITIALIZE THE ITT TABLES AND CLEAR SCREEN

  SUBROUTINE INITT

INITT2 254.7 INITIALIZE THE COLOR TABLES (NO DEANZA CALLS)

  SUBROUTINE INITT2

DZREAD 254.8 READS A LINE SEGMENT FROM SCREEN

  SUBROUTINE DZREAD(CHAN,IX,IY,LOCBUF,LENVAL)
                          CHAN IS IMAGE CHANNEL NUMBER (0-3)
                          IX IS X STARTING POSITION (0-511)
                          IY IS LINE NUMBER  (0-511)
                              (0 IS BOTTOM LINE)
                          LOCBUF IS THE LOC OF  UP TO 256 WORD BUFFER WITH
                          PIXEL INFO.

FUN406 255 SAVE/RESTORE ITEMS TO/FROM DISC

  SUBROUTINE FUN406

SAVE 255.1 SAVE AN ITEM

  SUBROUTINE SAVE(NOAREA)
  ALL DISC FILES ARE IDENTIFIED BY A FORMAT NUMBER CONTAINED IN THE FIRST
```

```
  FORTRAN READABLE INTEGER IN THE FILE (I12 FORMAT).
  THE INFORMATION AFTER THE FORMAT DEPENDS ON THE FORMAT AS FOLLOWS:
    FORMAT=1 - COMPRESSED NODES
      - IARTB INFORMATION - 12 VALUES FROM IARTB IN I12 FORMAT
      - NODE VALUES COMPRESSED USING 2 BITS PER NODE (4 NODES PER BYTE)
    FORMAT=2 - FULL 16 BIT FORMAT
      - IARTB INFORMATION - 12 VALUES FROM IARTB IN I12 FORMAT
      - NODE VALUES, INCLUDING ALL 16 BITS OF INFORMATION FROM IARVTB
        (1 NODE PER 16 BIT WORD)
    FORMAT=3 - ORIGINAL ASCII FORMAT
      - IARTB INFORMATION - 12 VALUES FROM IARTB IN I12 FORMAT
      - NODE VALUES, ONE ASCII VALUE (0,1,2 OR 9 FOR TERMINATION) PER NODE.
        FORTRAN READABLE, BLOCKED 80 CHARACTERS PER RECORD.

RESTOR 255.2 RESTORE ITEM FROM DISC

  SUBROUTINE RESTOR(IARNUM)
  ALL DISC FILES ARE IDENTIFIED BY A FORMAT NUMBER CONTAINED IN THE FIRST
  FORTRAN READABLE INTEGER IN THE FILE (I12 FORMAT).
  THE INFORMATION AFTER THE FORMAT DEPENDS ON THE FORMAT AS FOLLOWS:

SAVE2 255.3 SAVE AN ITEM (OLD FORMAT ONLY)

  SUBROUTINE SAVE2(NOAREA)

RESTR2 255.4 RESTORE ITEM FROM DISC (OLD FORMAT ONLY)

  SUBROUTINE RESTR2(IARNUM)

SAVE3 255.5 SAVE AN ITEM (SPECIFY FORMAT AND FILE)

  SUBROUTINE SAVE3(IFORMT,INAME,NOAREA)
  THIS SUBROUTINE WILL SAVE AN OBJECT IN A DISC FILE. IN THIS VERSION,
  THE FORMAT NUMBER AND FILE NAME IS SPECIFIED BY THE CALLING PROGRAM.
  THE ARGUMENTS ARE AS FOLLOWS:
    IFORMT - FORMAT NUMBER (1, 2, OR 3)
    INAME - ASCII CHARACTER STRING CONTAINING THE FILE NAME
    NOAREA - ITEM NUMBER OF THE OBJECT
  ALL DISC FILES ARE IDENTIFIED BY A FORMAT NUMBER CONTAINED IN THE FIRST
  FORTRAN READABLE INTEGER IN THE FILE (I12 FORMAT).
  THE INFORMATION AFTER THE FORMAT DEPENDS ON THE FORMAT AS FOLLOWS:

DZ3D 256 DISPLAY 3-D OBJECTS ON DEANZA

  SUBROUTINE DZ3D(IOPT,IOPTU,IFUNCT,IARARY,IAREAS,IPLANX,IPLANY,
      IPLANZ,ANG1,ANG2,ANG3,SCALE,IDIST)
  THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP
  TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
       IOPT - OPTION DESIRED:
                1 - CALLS FOR DIRECT DISPLAY ON DEANZA
```

```
                2 - CALLS PARAL2 FOR USE OF SUBPIXEL ARRAY
                3 - CALL PARAL5 FOR B AND W SUBPIXEL DISPLAY
        IFUNCT - DESIRED FUNCTION
                  1 - VIEW FROM INFINITE DISTANCE
                  2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
        IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                 TO BE DISPLAYED (IARTB POINTERS)
        IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
        IPLANX - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN X
        IPLANY - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN Y
        ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE
              TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2
              OF FROM 0 TO 90 DEGREES).
        IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE
                VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)

PARAL2 257 DRAW A PARALLELOGRAM (SUB-PIXEL)

  SUBROUTINE PARAL2(ITEM,ISIDE,IVERTX)
  THIS SUBROUTINE WILL PUT A PARALLELOGRAM IN THE SUBPIXEL ARRAY. THE
  ARGUMENTS ARE DEFINED AS FOLLOWS:
    ITEM - THE NUMBER OF THE OBJECT BEING DISPLAYED (1 TO 10)
    ISIDE - THE SIDE BEING DISPLAYED (1 TO 3)
    IVERTX - AN ARRAY (4,3) SPECIFYING THE X,Y LOCATIONS OF THE FOUR
             CORNERS OF THE PARALLELOGRAM IN CLOCKWISE ORDER BEGINNING
             WITH THE POINT WITH THE LOWEST Y VALUE (0<VALUE<511*1024)

SUBPIX 258 PRESENT A SUBPIXEL VIEW OF AN OBJECT

  SUBROUTINE SUBPIX(IOPT2,IOPTU,IFUNCT,IARARY,IAREAS,IPLANX,IPLANY,
      IPLANZ,ANG1,ANG2,ANG3,SCALE,IDIST,KORIGX,KORIGY,KDELX,KDELY,
      LEVSUB,IWINDX,IWINDY,IPLANE)
  THE ARGUMENTS ARE AS FOLLOWS:
        IOPT2 - OPTION VALUE. NORMALLY 0. IF 1, LEVSUB IS SET TO 2, THE IMAGE
                IS MOVED TO THE TOP QUARTER OF THE SCREEN AND THE FIRST 3
                SCANS (BEFORE INTEGRATION) ARE PRESENTED IN THE LOWER 3/4 OF
                THE SCREEN.
        IFUNCT - DESIRED FUNCTION
                  1 - VIEW FROM INFINITE DISTANCE
                  2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
        IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                 TO BE DISPLAYED (IARTB POINTERS)
        IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
        IPLANX - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN X
        IPLANY - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN Y
        ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE
              TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2
              OF FROM 0 TO 90 DEGREES).
        IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE
                VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
```

KORIGX - X VALUE OF THE BOTTOM LEFT PIXEL OF THE WINDOW TO BE PROCESSED (0 TO 511) (LOCATION ON FULL SCREEN OF THIS OBJECT OR OBJECT SPACE WINDOW)
KORIGY - SAME FOR Y
KDELX - NUMBER OF PIXELS IN THE WINDOW IN THE X DIRECTION
KDELY - SAME FOR Y
LEVSUB - NUMBER OF LEVELS BELOW THE PIXEL LEVEL AT WHICH TO PERFORM THE VIEW CONVERSION.IF LEVSUB=0, WE WORK AT THE PIXEL LEVEL. IF 1, EACH PIXEL IS REPRESENTED BY 4 VALUES (2 BY 2 ARRAY). IF 2, EACH IS PROCESSED AS 16 POINTS (4 BY 4), AND SO ON.
IWINDX - THE X LOCATION ON THE DEANZA DISPLAY (0 TO 511) OF THE BOTTOM LEFT HAND CORNER OF THE WINDOW WHEN DISPLAYED (SO CAN PUT MULTIPLE VIEWS ON THE SCREEN AT THE SAME TIME). (SCREEN WINDOW)
IWINDY - SAME BUT FOR Y.

SHEL3D 259 CREATE A VISIBLE SHELL VOLUME

SUBROUTINE SHEL3D(NOAREA,IARNUM,IHEADR)
THIS SUBROUTINE WILL GENERATE A NEW VOLUME ENCODED BLOB WITH ONLY THE POSSIBLY VISIBLE CUBES OF AN ITEM. AT THIS TIME, THIS MEANS VISIBLE TO AN OBSERVER LOCATED SOMEWHERE BETWEEN 0 AND 90 DEGREES IN THE VIEW ANGLES.
THE ARGUMENTS ARE AS FOLLOWS:
NOAREA - THE IARTB POINTER OF THE ITEM
IARNUM - THE NEW AREA IARTB POINTER
IHEADR - HEADER OF THE NEW AREA
NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES.

ILLUM 260 ALLOW THE MOVEMENT OF THE LIGHT SOURCE TO CHANGE THE ILLUMINATION

SUBROUTINE ILLUM(ANG1,ANG2)
ANG1 AND ANG2 ARE THE TWO ANGLES (RADIANS) FROM THE VIEW DIRECTION OF THE SOURCE OF ILLUMINATION (+ OR - PI/2)
ANG1 IS IN THE X-Z PLANE. ANG2 IS IN THE X-Y PLANE.

FUN407 261 ENTER AN AREA

SUBROUTINE FUN407

FUN7A 262 FUNCTION KEY 7 - DISPLAY A CHAIN OR AREA

SUBROUTINE FUN7A

FUN409 263 TRANSLATE (MOVE) AN AREA

SUBROUTINE FUN409

FUN408 264 PRINT SYSTEM TABLES ON THE SCREEN

SUBROUTINE FUN408

TOPT$$ 264.1 UTILITY FOR FUN408

SUBROUTINE TOPT$$(IOPT)
THIS SUBROUTINE ENTERS GRAPHICS MODE, CLEARS THE MENU AND MESSAGE FIELDS
AND THEN ALLOWS THE USER TO SELECT ONE OF THE FOLLOWING FROM THE MENU:

1 - / MORE /
2 - /NEW TABLE/
3 - / CLEAR /
4 - / ABORT /

FOR OPTION 3, THE A/N SCREEN IS CLEARED AND THE OPTIONS ARE REPEATED. FOR
THE OTHERS, THE OPTION VALUE IS RETURNED.

SPHERE 265 GENERATE SPHERE

SUBROUTINE SPHERE(ITEMNO,IHEADR, ENTX,CENTY,CENTZ,RADIUS)
THIS SUBROUTINE WILL GENERATE A SPHERE BY GENERATING CHAINS FOR
CIRCLES, CONVERTING THEM INTO SLICES AND ADDING THEM (UNION)
TOGETHER TO FORM A SPHERE.
IT IS GENERATED AT THE CURRENT LEVEL. THE CENTER AND RADIUS ARE
MODIFIED TO CORRESPOND TO GRID POINTS.
THE PARAMETERS ARE AS FOLLOWS:

ITEMNO - ITEM NUMBER (POINTER TO IARTB) (RETURNED)
IHEADR - HEADER (SPECIFIED)
(CENTX,CENTY,CENTZ) - CENTER OF CIRCLE
RADIUS - RADIUS OF CIRCLE

FUN412 266 GENERATE A CIRCLE CHAIN OR SHPERE

SUBROUTINE FUN412

ILLJS 267 ILLUMINATION PARAMETER CHANGE BY JOYSTICK

SUBROUTINE ILLJS
THIS SUBROUTINE ALLOWS THE USER TO CHANGE THE ILLUMINATION ON AN
OBJECT DISPLAYED WHICH USES THE ITT'S TO DEFINE THE COLOR. (THIS
CANNOT BE USED FOR SUBPIXEL USE INTERACTIVELY.) THE TWO CURSORS
ARE USED. CURSOR #1 CONTROLS THE ILLUMINATION POINT. THE CENTER
OF THE SCREEN IS THE VIEWPOINT OF THE UNIVERSE. THE SECOND CURSOR
IS USED TO SET THE EXPONENT (X DIRECTION) AND THE OFFSET (Y DIRECTION)
IN THE COLOR AND ILLUMINATION DEFINITIONS.

ENDARV 268 LOCATE THE END OF THE AREA VALUE TABLE (IARVTB)

SUBROUTINE ENDARV(IEND)
THIS SUBROUTINE RETURNS THE LOCATION OF THE END LOCATION
OF THE AREA VALUE TABLE (POINTER TO THE FIRST -2 VALUE).

ENDCHV 268.1 LOACTE THE END OF THE CHAIN VALUE TABLE (ICHVTB)

```
  SUBROUTINE ENDCHV(IEND)
  THIS SUBROUTINE RETURNS THE LOCATION OF THE END LOCATION
  OF THE CHAIN VALUE TABLE (POINTER TO THE FIRST -2 VALUE).

MOVE3D 269 MOVE (TRANSLATE) A 3D AREA

  SUBROUTINE MOVE3D(ICALL,NOAREA,MOVEX,MOVEY,MOVEZ,IHEADR,
      IARNUM,MINLEV)
  THIS SUBROUTINE WILL CONVERT AN OCTREE ENCODED BLOB INTO A SECOND WHICH IS
  A TRANSLATED VERSION OF THE FIRST. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      NOAREA - THE IARTB POINTER OF THE AREA TO BE TRANSLATED
      MOVEX - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE X
              DIRECTION (POSITIVE NUMBER ONLY)
      MOVEY - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE Y
              DIRECTION (POSITIVE NUMBER ONLY)
      MOVEZ - SAME FOR Z
      IARNUM - THE IARTB POINTER FOR THE NEW (TRANSLATED) AREA
      IHEADR - THE HEADER VALUE FOR THE NEW (TRANSLATED) AREA
      MINLEV - MINIMUM LEVEL OF NEW AREA (BECAUSE THE NEW UNIVERSE PROBABLY
               WILL NOT COMPLETELY ALLIGN WITH THE OLD UNIVERSE, THE PROCESS
               MAY NEVER CONVERGE FOR SOME SQUARES). TYPICALLY SET TO LEVEL
               OF OLD UNIVERSE OR ONE OR TWO BELOW.
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

STACK 270 STACK AN AREA FROM Z1 TO Z2

  SUBROUTINE STACK(NOAREA,ITEMNO,IHEADR,Z1,Z2,IOFFST)
    NOAREA - AREA NUMBER
    ITEMNO - NUMBER OF THE 3D ITEM GENERATED (RETURNED)
    IHEADR - HEADER FOR THE NEW ITEM
    Z1 - VALUE FOR FIRST SLICE (-4.5 TO 4.5)
    Z2 - VALUE FOR THE LAST SLICE (-4.5 TO 4.5)
    IOFFST - OFFSET WILL BE ADDED TO SLICE VALUE IN ADDITION TO SINGLE
             STEP IN Z DIRECTION (NORMALLY SHOULD BE SET TO 0 )
  SLICES ARE GENERATED AT THE CURRENT LEVEL. THE Z VALUES ARE USED TO
  GENERATE THE SPECIFIC BEGINNING AND ENDING SLICES AT THE CURRENT LEVEL.

STACK2 270.1 STACK AN AREA FROM NZ1 TO NZ2 (LEVEL 0)

  SUBROUTINE STACK2(NOAREA,ITEMNO,IHEADR,NZ1,NZ2,IOFFST)
    NOAREA - AREA NUMBER
    ITEMNO - NUMBER OF THE 3D ITEM GENERATED (RETURNED)
    IHEADR - HEADER FOR THE NEW ITEM
    NZ1 - VALUE FOR FIRST SLICE (LEVEL 0)
    NZ2 - VALUE FOR THE LAST SLICE (LEVEL 0)
    IOFFST - OFFSET WILL BE ADDED TO SLICE VALUE IN ADDITION TO SINGLE
             STEP IN Z DIRECTION (NORMALLY SHOULD BE SET TO 0 )
  SLICES ARE GENERATED AT THE CURRENT LEVEL. THE Z VALUES ARE USED TO
  GENERATE THE SPECIFIC BEGINNING AND ENDING SLICES AT THE CURRENT LEVEL.
```

ROT3D 271 ROTATE AN OBJECT - (0 TO PI/2 (0 TO 90 DEG))

```
  SUBROUTINE ROT3D(ICALL,NOAREA,MOVEX,MOVEY,ANG,IARNUM,IHEADR,
      MINLEV)
  THIS SUBROUTINE WILL CONVERT AN OCTREE ENCODED OBJECT INTO A SECOND WHICH
  IS A ROTATED VERSION OF THE FIRST. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      NOOBJECT - THE IARTB POINTER OF THE AREA TO BE ROTATED
      MOVEX - THE LEVEL 0 VALUE AT WHICH THE OBJECT IS TO BE ROTATED (IN
              THE X DIRECTION) (POSITIVE NUMBER ONLY)
      MOVEY - THE LEVEL 0 VALUE AT WHICH THE OBJECT IS TO BE ROTATED (IN
              THE Y DIRECTION) (POSITIVE NUMBER ONLY)
      ANG - ROTATION ANGLE (RADIANS) OF THE NEW UNIVERSE (OBJECTS MOVED BY
            -ANG IN THE NEW UNIVERSE). THIS IS A FLOATING POINT VALUE
            BETWEEN 0 AND PI/2.
      IARNUM - THE IARTB POINTER FOR THE NEW (ROTATED) OBJECT
      IHEADR - THE HEADER VALUE FOR THE NEW (ROTATED) OJECT
      MINLEV - MINIMUM LEVEL OF NEW OBJECT (BECAUSE THE NEW UNIVERSE
               PROBABLY WILL NOT COMPLETELY ALLIGN WITH THE OLD UNIVERSE,
               THE PROCESS MAY NEVER CONVERGE FOR SOME SQUARES). TYPICALLY
               SET TO LEVEL OF OLD UNIVERSE OR ONE OR TWO BELOW.
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES
```

DZ3P 272 3-D DEANZA DISPLAY (WITH PERSPECTIVE)

```
  SUBROUTINE DZ3P(IOPT,IOPTU,IFUNCT,IARARY,IAREAS,IPLANE,
      ANG1,ANG2,ANG3,SCALE,IDIST2,IMOVEX,IMOVEY,NSUBUN,NSUBVW)
  THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP
  TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:
       IOPT - CALL OPTION
           1 - DIRECT DISPLAY ON DEANZA (USING PARAL3)
           2 - PARAL4 DISPLAY INTO SUBPIXEL ARRAY
           3 - CALLS PARAL6 FOR B AND W USE OF SUBPIXEL DISPLAY
       IOPTU - IF 1 DISPLAY UNIVERSE (DON'T DISPLAY IF 0)
       IFUNCT - DESIRED FUNCTION
                  1 - VIEW FROM INFINITE DISTANCE
                  2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
       IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS
                TO BE DISPLAYED (IARTB POINTERS)
       IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
       IPLANE - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE
                (1=X,2=Y,3=Z)
       ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE
             TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2
             OF FROM 0 TO 90 DEGREES).
       IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE
               VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
       IMOVEX - MOVE THE CENTER OF THE VIEWPLANE BY THIS DITANCE (LEVEL
                0 UNITS IN THE X DIRECTION (SCREEN COORDINATES)
```

```
      IMOVEY - SAME FOR Y

PARAL3 273 DRAW A QUADRALATERAL ON THE DEANZA DISPLAY (PERSPECTIVE)

  SUBROUTINE PARAL3(ITEM,ISIDE2,IVERTX)
  THIS SUBROUTINE WILL PUT A QUADRALATERAL ON THE DEANZA DISPLAY. THE
  ARGUMENTS ARE DEFINED AS FOLLOWS:
    ITEM - THE NUMBER OF THE OBJECT BEING DISPLAYED (1 TO 10)
    ISIDE - THE SIDE BEING DISPLAYED (1 TO 3)
    IVERTX - AN ARRAY (4,2) SPECIFYING THE X,Y LOCATIONS OF THE FOUR
             CORNERS OF THE QUADRALATERAL.
             NOTE: THEY DO NOT HAVE TO BE IN CLOCKWISE ORDER (THIS PROGRAM
                   CHECKS THIS).

PARAL4 274 DRAW A QUADRALATERAL  (SUB-PIXEL) (PERSPECTIVE)

  SUBROUTINE PARAL4(ITEM,ISIDE2,IVERTX)
  THIS SUBROUTINE WILL PUT A QUADRALATERAL IN THE SUBPIXEL ARRAY. THE
  ARGUMENTS ARE DEFINED AS FOLLOWS:
    ITEM - THE NUMBER OF THE OBJECT BEING DISPLAYED (1 TO 10)
    ISIDE - THE SIDE BEING DISPLAYED (1 TO 6)
    IVERTX - AN ARRAY (4,3) SPECIFYING THE X,Y LOCATIONS OF THE FOUR
             CORNERS OF THE QUADRALATERAL
             NOTE: THEY DON'T HAVE TO BE IN CLOCKWISE ORDER (SEE PARAL3)

SCAL3D 275 SCALE A 3D AREA

  SUBROUTINE SCAL3D(ICALL,NOAREA,MOVEX,MOVEY,MOVEZ,IHEADR,
      IARNUM,MINLEV,SCALEX,SCALEY,SCALEZ)
  THIS SUBROUTINE WILL CONVERT AN OCTREE ENCODED BLOB INTO A SECOND WHICH IS
  A SCALED  VERSION OF THE FIRST. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      NOAREA - THE IARTB POINTER OF THE AREA TO BE TRANSLATED
      MOVEX - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE X
              DIRECTION (POSITIVE NUMBER ONLY)
      MOVEY - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE Y
              DIRECTION (POSITIVE NUMBER ONLY)
      MOVEZ - SAME FOR Z
      IARNUM - THE IARTB POINTER FOR THE NEW (TRANSLATED) AREA
      IHEADR - THE HEADER VALUE FOR THE NEW (TRANSLATED) AREA
      MINLEV - MINIMUM LEVEL OF NEW AREA (BECAUSE THE NEW UNIVERSE PROBABLY
               WILL NOT COMPLETELY ALLIGN WITH THE OLD UNIVERSE, THE PROCESS
               MAY NEVER CONVERGE FOR SOME SQUARES). TYPICALLY SET TO LEVEL
               OF OLD UNIVERSE OR ONE OR TWO BELOW.
      SCALEX - SCALE FACTOR IN THE X DIRECTION (1.0<SCALEX<2.0)
      SCALEY - SAME FOR Y
      SCALEZ - SAME FOR Z
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

FUN413 276 SCALE AN OBJECT
```

```
  SUBROUTINE FUN413

SCAL2D 277 SCALE A 2D AREA

  SUBROUTINE SCAL2D(ICALL,NOAREA,MOVEX,MOVEY,IHEADR,
      IARNUM,MINLEV,SCALEX,SCALEY)
  THIS SUBROUTINE WILL CONVERT A QUADTREE ENCODED BLOB INTO A SECOND WHICH
  IS A SCALED VERSION OF THE FIRST. THE ARGUMENTS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      NOAREA - THE IARTB POINTER OF THE AREA TO BE TRANSLATED
      MOVEX - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE X
              DIRECTION (POSITIVE NUMBER ONLY)
      MOVEY - THE LEVEL 0 VALUE BY WHICH THE AREA IS TO BE MOVED IN THE Y
              DIRECTION (POSITIVE NUMBER ONLY)
      IARNUM - THE IARTB POINTER FOR THE NEW (TRANSLATED) AREA
      IHEADR - THE HEADER VALUE FOR THE NEW (TRANSLATED) AREA
      MINLEV - MINIMUM LEVEL OF NEW AREA (BECAUSE THE NEW UNIVERSE PROBABLY
               WILL NOT COMPLETELY ALLIGN WITH THE OLD UNIVERSE, THE PROCESS
               MAY NEVER CONVERGE FOR SOME SQUARES). TYPICALLY SET TO LEVEL
               OF OLD UNIVERSE OR ONE OR TWO BELOW.
      SCALEX - SCALE FACTOR IN THE X DIRECTION (1.0<SCALEX<2.0)
      SCALEY - SAME FOR Y
  NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES

TAG 278 TAG AN OCTREE ITEM (CREATE NEW ITEM)

  SUBROUTINE TAG(NOAREA,IARNUM,IHEADR,MINLEV)
  THIS SUBROUTINE WILL GENERATE A NEW OCTREE ENCODED ITEM WITH ONLY THE
  POSSIBLY VISIBLE CUBES OF AN ITEM. AT THIS TIME, THIS MEANS VISIBLE TO AN
  OBSERVER LOCATED SOMEWHERE BETWEEN 0 AND 90 DEGREES IN THE VIEW ANGLES.
  IN ADDITION, THE NEW ITEM IS TAGGED IN THE FOLLOWING FORMAT:
    -- 16 --
    13 14 15   Y=+
    -- 12 --
                                ---------> Z
    -- 10 11                    !
    08 02 09                    !
    05 06 07   Y=0              !
                                !
    -- -- --                    X
    -- 03 04   Y=-
    -- 01 --
  THE ARGUMENTS ARE AS FOLLOWS:
       NOAREA - THE IARTB POINTER OF THE ITEM
       IARNUM - THE NEW AREA IARTB POINTER (RETURNED)
                NOTE: IARNUM IS RETURNED AS A 0 IF NO FREE SLOTS ARE FOUND
                      OR THE END OF VALUE TABLE IARVTB IS REACHED.
       IHEADR - HEADER OF THE NEW AREA (NOTE: BIT 3 WILL BE SET)
       MINLEV - LEVEL AT WHICH ALL PARTIALS ARE CONVERTED TO FULLS
```

THE NODE VALUES PLACED IN THE TABLE (IARTB) WILL BE TAGGED VALUES FOR THE NODE. BIT POSITION N (LS BIT IS BIT 1) CORRESPONDS TO THE ABOVE NEIGHBORING NUMBERING. A 1 BIT INDICATES THAT THE NEIGHBOR IS FULL AND A 0 IS EMPTY. THE CENTER NODE IS IN BIT POSITION 2. THIS ALLOWS THE VALUES OF EMPTY AND PARTIAL TO REMAIN THE SAME (0 AD 1):

0...00 - EMPTY
0...01 - PARTIAL
-...1- - FULL WITH STATUS OF 15 NEIGHBORS INDICATED IN REMAINING BITS

NOTE: THE SCAN TABLE (ISCAN) IS USED TO HOLD ADDRESSES.

VIEWTG 279 DISPLAY THE SQUARES OF MULTIPLE AREAS IN SORTED ORDER

SUBROUTINE VIEWTG(ICALL,IOPTU,IFUNCT,IARARY,IAROPT,IAREAS,IPLANX, IPLANY,IPLANZ,ANG1,ANG2,ANG3,SCALE,IDIST,NSUBUN,NSUBVW,NSUBIF)

THIS SUBROUTINE WILL DISPLAY THE (2 AND 3) EDGES OF THE SQUARES OF UP TO 10 AREA ENCODED ITEMS IN SORTED ORDER. THE ARGUMENTS ARE AS FOLLOWS:

ICALL - CALL NUMBER
IOPTU - UNIVERSE DISPLAY OPTION (DO NOT DISPLAY EDGE OF UNIVERSE IF IOPTU=0
IFUNCT - DESIRED FUNCTION
1 - VIEW FROM INFINITE DISTANCE
2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS TO BE DISPLAYED (IARTB POINTERS)
IAROPT - INTEGER VECTOR OF 10 ELEMENTS. HOLDS OPTS FOR EACH ELEMENT:
0 - NO DISPLAY OR INTERFERENCE CHECK (USED AS PLACEHOLDER FOR OBJECTS SO AS TO MAINTAIN COLOR FROM VIEW TO VIEW)
1 - DISPLAY
2 - NO DISPLAY, BUT PUT IN FLASHING SUBPICTURE IF INTERSECTS TYPE 1
IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
IPLANX - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN X
IPLANY - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN Y
ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2 OF FROM 0 TO 90 DEGREES).
IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
NSUBUN - SUBPICTURE NUMBER OF UNIVERSE (0 IF NOT USED)
NSUBVW - SUBPICTURE NUMBER OF VIEW (RETURNED). NOTE: IF ONLY ONE OBJECT DISPLAYED, IT HAS A TYPE OF AREA IN THE SUBPICTURE TABLE AND THE OBJECT CAN THEREFORE BE DELETED LATER THROUGH THE SUBPICTURE.
NSUBIF - SUBPICTURE NUMBER OF INTERFERENCE (FLASHING) (0 IF NOT USED)

RECTCH 280 RECTANGLE CHAIN INPUT

SUBROUTINE RECTCH(ICALL,ICH,IHEADR,STARTX,STARTY,DELX,DELY)

THIS SUBROUTINE IS USED TO GENERATE A RECTANGULAR CHAIN AND PUT IT INTO THE TABLES.

```
  THE PARAMETERS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      ICH - CHAIN NUMBER (RETURNED TO CALLING PROGRAM)
      IHEADR - HEADER FOR NEW CHAIN
      STARTX,STARTY - LOCATION OF BOTTOM LEFT CORNER OF RECTANGLE (-4.5 TO
                      4.5)
      DELX,DELY - WIDTH AND HEIGTH OF RECTANGLE (-4.5 - 4.5)

BLOCK 281 CREATE A 3-D OCTREE BLOCK

  SUBROUTINE BLOCK(IARNUM,LEVSEL,ORIGX,ORIGY,ORIGZ,DELX,DELY,DELZ)
  THIS SUBROUTINE WILL GENERATE A 3-D BLOCK IN OCTREE FORMAT FROM THE ORIGIN
  LOCATION IN 3-D AND THE THREE EDGE SIZES. IT IS GENERATED AT A SPECIFIED
  LEVEL.
    IARNUM - THE NEW OCTREE OBJECT NUMBER (RETURNED)
    LEVSEL - DESIRED LEVEL
    ORIGX,ORIGY,ORIGZ - ORIGIN (LOWEST VALUE IN EACH DIMENSION) (-4.5 - 4.5)
    DELX,DELY,DELZ - EDGE SIZES OF BLOCK (-4.5 TO 4.5)

DELAR 283 DELETE AREA

  SUBROUTINE DELAR(IARNUM)
  THIS SUBROUTINE WILL DELETE AN AREA OR OCTREE ENCODED OBJECT.
  FIRST THE SUBPICTURE TABLE IS SEARCHED. IF A CORRESPONDING
  SUBPICTURE EXISTS, DELSUB IS CALLED WITH IT. IF NOT A
  DUMMY ONE IS GENERATED AND DELSUB IS CALLED.

DELCH 283.1 DELETE CHAIN

  SUBROUTINE DELCH(ICH)
  THIS SUBROUTINE WILL DELETE A CHAIN.
  FIRST THE SUBPICTURE TABLE IS SEARCHED. IF A CORRESPONDING
  SUBPICTURE EXISTS, DELSUB IS CALLED WITH IT. IF NOT A
  DUMMY ONE IS GENERATED AND DELSUB IS CALLED.

DARMNU 284 DISPLAY AREA IN MENU FIELD

  SUBROUTINE DARMNU
  THE NUMBERS OF DEFINED OBJECTS IN THE AREA TABLES ARE
  PUT INTO THE MENU FIELD. THE ITYPE NUMBER AFTER A
  SELECT IS CALLED (BY THE CALLING PROGRAM) WILL BE:
    IOBJNO=NSUB-MENU+1

QUADCH 285 QUADRALATERAL CHAIN INPUT

  SUBROUTINE QUADCH(ICALL,ICH,IHEADR,IVERTX)
  THE PARAMETERS ARE DEFINED AS FOLLOWS:
      ICALL - CALL NUMBER
      ICH - CHAIN NUMBER (RETURNED TO CALLING PROGRAM)
      IHEADR - HEADER FOR NEW CHAIN
```

IVERTX(A,B) - THE FOUR VERTEX POINTS (LEVEL 0 VALUES)
A - POINT (1 TO 4)
B - DIMENSION (1=X, 2=Y)
NOTE: THE IVERTX VALUES RETURNED TO THE CALLING PROGRAM ARE STILL AT LEVEL 0 BUT HAVE BEEN TO NORMALIZED TO THE CLOSEST GRID POINT AT THE CURRENT LEVEL (NLINES IS USED). FOR EX., IF NLINES=8, AN INPUT VALUE OF 18 OR 17 WOULD BE RETURNED AS 16.

POLYCH 286 POLYGONAL CHAIN INPUT

SUBROUTINE POLYCH(ICALL,ICH,IHEADR,IVERTX,NOPTS,IERR)

THE PARAMETERS ARE DEFINED AS FOLLOWS:
ICALL - CALL NUMBER
ICH - CHAIN NUMBER (RETURNED TO CALLING PROGRAM)
IHEADR - HEADER FOR NEW CHAIN
IVERTX(A,B) - THE VERTEX POINTS (LEVEL 0 VALUES)
A - DIMENSION (1=X, 2=Y)
B - VERTEX POINT
NOPTS - NUMBER OF POINTS IN IVERTX ARRAY
IERR - ERROR RETURN VALUE (0 IF SUCCESSFUL, 1 IF ICHVTB OVERFLOW, 2 IF ICHTB OVERFLOW)
NOTE: THE IVERTX VALUES RETURNED TO THE CALLING PROGRAM ARE STILL AT LEVEL 0 BUT HAVE BEEN NORMALIZED TO THE CLOSEST GRID POINT AT THE CURRENT LEVEL (NLINES IS USED). FOR EX., IF NLINES=8, AN INPUT VALUE OF 18 OR 17 WOULD BE RETURNED AS 16.

LINKS 287 GENERATE CHAIN LINKS BETWEEN TWO POINTS

SUBROUTINE LINKS(IDELX,IDELY,IPOINT,IERR)
THE CHAIN LINKS BETWEEN TWO POINTS DEFINED BY THE TWO DELTA VALUES ARE GENERATED. THEY ARE AT THE LEVEL OF THE INPUT DELTA VALUES.
THE ARGUMENTS ARE AS FOLLOWS:
IDELX - DELTA VALUE IN X DIRECTION
IDELY - SAME IN Y
IPOINT - POINTER INTO ICHVTB TABLE FOR THE FIRST LINK (RETURNED AS THE LAST LINK LOCATION + 1)
IERR - ERROR FLAG (0 IF NO ERROR, 1 IF HIT END OF ICHVTB). IF HIT END OF ICHVTB, THE LAST ELEMENT IS SET TO -1 AND IPOINT POINTS TO IT (THE LAST ELEMENT IN THE TABLE).

PRISM 288 GENERATE A PRISM FROM A LIST OF VERTICES

SUBROUTINE PRISM(IARNUM,IHEADR,IVERTX,NOPTS,IZ1,IZ2)
THIS SUBROUTINE ACCEPTS A SERIES OF VERTEX POINTS AND TWO Z VALUES AND CREATES A PRISM (DEPTH IS IN THE Z DIMENSION). THE ARGUMENTS ARE

AS FOLLOWS:

IARNUM - NEW ITEM NUMBER (RETURNED)
NOTE: IARNUM IS RETURNED AS 0 IF UNSUCCESSFUL (ERROR).

IHEADR - HEADER VALUE FOR NEW ITEM

IVERTX(A,B) - 2 DIMENSIONAL ARRAY OF VERTICES (LEVEL 0 VALUES)
A - DIMENSION (1=X, 2=Y)
B - VERTEX POINT
NOTE: THE VERTEX POINTS SHOULD BE IN CLOCKWISE ORDER. THE LAST POINT WILL BE CLOSED TO THE FIRST.

NOPTS - NUMBER OF POINTS IN IVERTX ARRAY

IZ1 - LEVEL 0 VALUE OF FACE IN Z

IZ2 - LEVEL 0 VALUE OF FACE IN Z (SECOND FACE)
NOTE: IX1 AND IZ2 ARE REVERSED IF IN THE WRONG ORDER.

FREEIT 289 FIND A FREE ITEM

SUBROUTINE FREEIT(IHEADR,IARNUM,IARVPT,IERR)

NOTE!! IF THIS OBJECT IS TO BE PASSED TO ANOTHER SUBROUTINE, ENDARV CANNOT BE USED FOR IT. GREAT CONFUSION WILL RESULT BECAUSE THIS SLOT WILL ALREADY BE TAKEN!

THIS SUBROUTINE WILL FIND A FREE SLOT IN THE OBJECT TABLE IARTB. THE HEADER IS SET TO THE SPECIFIED VALUE, THE LEVEL IS SET TO THE CURRENT LEVEL AND THE FIRST FREE VALUE LOCATION IN IARVTB IS PUT INTO IARTB AND RETURNED. OTHER LOCATIONS ARE CLEARED. IF NO LOCATIONS ARE FOUND, AN ERROR MESSAGE IS GENERATED AND AN ERROR VALUE IS RETURNED. NOTE: THE AREA IS RESERVED. DO NOT USE THIS SUBROUTINE TO SIMPLY DETERMINE IF ANY SLOTS ARE AVAILABLE (USE FUNCTION NUMFRE).

NOTE!!!!! - THE IARTB TABLE VALUES INDICATING A VALID OBJECT AND POINTING TO THE NEW BEGINNING IN THE IARVTB TABLE ARE NOW SET. DO NOT USE ENDARV AGAIN TO FIND THE BEGINNING OF THE VALUE TABLE (IT WILL GET ALL CONFUSED).

THE ARGUMENTS ARE AS FOLLOWS:

IHEADR - HEADER VALUE FOR NEW OBJECT

IARNUM - NEW AREA OBJECT NUMBER (RETURNED)

IARVPT - LOCATION IN IARVTB OF FIRST VALUE (RETURNED).
NOTE: SOME SUBROUTINES REQUIRE THIS POINTER TO BEGIN AT THE FIRST LOCATION -1.

IERR - ERROR CODE (RETURNED)
0 - NO ERROR
1 - NO FREE OBJECT LOCATION FOUND IN IARTB (ERROR MESSAGE GENERATED)
2 - AN ATTEMPT TO SPECIFY A HEADER OF 0 (ERROR MESSAGE GENERATED)

LEVEL0 290 CONVERT SCREEN VALUE TO NORMALIZED LEVEL 0 INTEGER

INTEGER FUNCTION LEVEL0(VAL)

THIS SUBROUTINE WILL CONVERT A SCREEN VALUE (REAL FROM -4.5 TO 4.5) INTO A LEVEL ZERO INTEGER WHICH IS NORMALIZED TO FALL ON A GRID POINT AT THE CURRENT LEVEL.

NEWLEV 291 SET NEW LEVEL

SUBROUTINE NEWLEV(N)

ROTUP 292 ROTATE AN OBJECT UP BY 90 DEG

SUBROUTINE ROTUP(NOAREA)
THIS SUBROUTINE WILL CREATE A NEW OBJECT WHICH IS THE 3-D OBJECT ROTATED 270 DEG ABOUT THE X AXIS (90 DEG UP AS LOOK AT SCREEN FROM -Z DIRECTION). OBJECTS IN THE X-Y PLANE ARE NOW IN THE X-Z PLANE. THE RETURNED OBJECT NUMBER WILL BE DIFFERENT (OLD OBJECT IS DESTROYED). IF THERE IS AN ERROR, NOAREA IS RETURNED AS 0.

NUMFRE 293 LOGICAL FUNCTION - TRUE IF THERE ARE N FREE ITEMS

STPATH 294 GENERATE STRAIGHT PATH OBJECT OF A CUTTER MOVEMENT

SUBROUTINE STPATH(IARNUM,IHEADR,IX1,IZ1,IX2,IZ2,IRADUS,IYLO,IYHI)
AN OBJECT IS GENERATED WHICH IS THE VOLUME SWEPT OUT BY A CYLINDRICAL CUTTER IN A STRAIGHT LINE MOVEMENT BETWEEN TWO POINTS IN THE X-Z PLANE. THE OBJECT IS A BLOCK. THE CUTTER SHOULD BE ADDED AT EACH END LATER. THE UPPER AND LOWER Y VALUES ARE SPECIFIED.
THE ARGUMENTS ARE AS FOLLOWS:
IARNUM - NEW OBJECT NUMBER (RETURNED AS 0 IF ERROR) (RETURNED)
IHEADR - HEADER OF NEW OBJECT
(IX1,IZ1), (IX2,IZ2) - THE TWO CENTER POINTS OF THE TWO TOOL POSITIONS IN THE X-Z PLANE
IRADIUS - RADIUS OF CYLINDER CUTTER (LEVEL 0 VALUE)
IYLO AND IYHI - LOWER AND UPPER Y VALUES (LEVEL 0 VALUES)

REDUCE 296 REDUCE A TREE

SUBROUTINE REDUCE(IARNUM,IEND)
THIS SUBROUTINE WILL REDUCE A TREE (REMOVE UNNEEDED NODES). IT IS USED AT THE END OF A TREE CREATION. THIS SUBROUTINE WILL FILL IN THE COUNT IN IARTB AND THE -1 AT THE END OF THE DATA VALUES.
THE ARGUMENTS ARE AS FOLLOWS:
IARNUM - NUMBER OF THE OBJECT (NOTE: THIS IS CHECKED TO MAKE SURE THAT IT IS IS THE LAST IN THE IARVTB TABLE. ERROR MESSAGE IF NOT.)
IEND - IARVTB POINTER TO LAST VALUE (A -1 IS PUT AT IARVTB+1)
NOTE: THE TREE MUST JUST HAVE BEEN CREATED AND THEREFORE BE AT THE END OF THE PART OF THE IARVTB TABLE IN USE. THE REDUCED PART OF THE TREE IS FILLED WITH -2'S.

PARAL5 297 DRAW A PARALLELOGRAM (SUB-PIXEL AND B AND W)

SUBROUTINE PARAL5(ITEM,ISIDE,IVERTX)
THIS SUBROUTINE WILL PUT A PARALLELOGRAM IN THE SUBPIXEL ARRAY. THE ARGUMENTS ARE DEFINED AS FOLLOWS:

ITEM - THE NUMBER OF THE OBJECT BEING DISPLAYED (1 TO 10)
ISIDE - THE SIDE BEING DISPLAYED (1 TO 3)
IVERTX - AN ARRAY (4,3) SPECIFYING THE X,Y LOCATIONS OF THE FOUR CORNERS OF THE PARALLELOGRAM IN CLOCKWISE ORDER BEGINNING WITH THE POINT WITH THE LOWEST Y VALUE (0<VALUE<511*1024)

SUBPX2 298 PRESENT A SUBPIXEL VIEW OF AN OBJECT (BLACK AND WHITE ONLY)

SUBROUTINE SUBPX2(IOPT2,IOPTU,IFUNCT,IARARY,IAREAS,IPLANX,IPLANY, IPLANZ,ANG1,ANG2,ANG3,SCALE,IDIST,KORIGX,KORIGY,KDELX,KDELY, LEVSUB,IWINDX,IWINDY,IPLANE)
THE ARGUMENTS ARE AS FOLLOWS:
IOPT2 - OPTION VALUE. NORMALLY 0. IF 1, LEVSUB IS SET TO 2, THE IMAGE IS MOVED TO THE TOP QUARTER OF THE SCREEN AND THE FIRST 3 SCANS (BEFORE INTEGRATION) ARE PRESENTED IN THE LOWER 3/4 OF THE SCREEN.
IFUNCT - DESIRED FUNCTION
1 - VIEW FROM INFINITE DISTANCE
2 - VIEW FROM DISTANCE IDIST (LEVEL 0 UNITS)
IARARY - INTEGER VECTOR OF 10 ELEMENTS. THIS HOLDS THE AREA NUMBERS TO BE DISPLAYED (IARTB POINTERS)
IAREAS - NUMBER OF AREAS TO BE DISPLAYED (1 TO 10)
IPLANX - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN X
IPLANY - LOCATION (LEVEL 0) OF THE CENTER OF THE VIEW PLANE IN Y
ANG - ANGLE (RADIANS) OF THE LINE FROM THE CENTER OF THE VIEW PLANE TO THE VIEWER, RELATIVE TO THE X AXIS (VALUE FROM 0 TO PI/2 OF FROM 0 TO 90 DEGREES).
IDIST - DISTANCE FROM THE CENTER OF THE PROJECTION VIEW PLANE TO THE VIEWER (SAVES COMPUTATION (LATER) IF IS POWER OF 2)
KORIGX - X VALUE OF THE BOTTOM LEFT PIXEL OF THE WINDOW TO BE PROCESSED (0 TO 511) (LOCATION ON FULL SCREEN OF THIS OBJECT OR OBJECT SPACE WINDOW)
KORIGY - SAME FOR Y
KDELX - NUMBER OF PIXELS IN THE WINDOW IN THE X DIRECTION
KDELY - SAME FOR Y
LEVSUB - NUMBER OF LEVELS BELOW THE PIXEL LEVEL AT WHICH TO PERFORM THE VIEW CONVERSION.IF LEVSUB=0, WE WORK AT THE PIXEL LEVEL. IF 1, EACH PIXEL IS REPRESENTED BY 4 VALUES (2 BY 2 ARRAY). IF 2, EACH IS PROCESSED AS 16 POINTS (4 BY 4), AND SO ON.
IWINDX - THE X LOCATION ON THE DEANZA DISPLAY (0 TO 511) OF THE BOTTOM LEFT HAND CORNER OF THE WINDOW WHEN DISPLAYED (SO CAN PUT MULTIPLE VIEWS ON THE SCREEN AT THE SAME TIME). (SCREEN WINDOW)
IWINDY - SAME BUT FOR Y.

PARAL6 299 DRAW A QUADRALATERAL (SUB-PIXEL) (PERSPECTIVE) (B AND W)

SUBROUTINE PARAL6(ITEM,ISIDE2,IVERTX)
THIS SUBROUTINE WILL PUT A QUADRALATERAL IN THE SUBPIXEL ARRAY. THE ARGUMENTS ARE DEFINED AS FOLLOWS:

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER IPL-TR-028	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Documentation Manual Graphics Package No. 1 (GP/1) and Program OCTREE		5. TYPE OF REPORT & PERIOD COVERED Technical 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Donald Meagher		8. CONTRACT OR GRANT NUMBER(s) N00014-82-K-0301
9. PERFORMING ORGANIZATION NAME AND ADDRESS Rensselaer Polytechnic Institute Troy, New York 12181		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research 800 North Quincy Street Arlington, VA 22217		12. REPORT DATE August 1982 13. NUMBER OF PAGES 151
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

"The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notices heron."

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer graphics
solid modeling
object manipulation algorithms
pattern recognition
octree representation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Graphics Package No. 1 (GP/1) is a general-purpose support package for graphics applications. It is written in Fortran and operates on a Prime computer systen with an Imlac 6220 graphics terminal. The user intreface is presented along with a sample work session. It is intended for users of systems based on GP/1. Programmer information is also provided for new applications.

Program OCTREE is a solid modeling package based on GP/1. It also requires a DeAnza IP5532 color raster graphics display for shaded putput (the system was developed fo- use at the Image Processing Laboratory at Rensselaer (over)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Polytechnic Institute. User and programming information is presented.